INF-3066

GAUHATI UNIVERSITY

Centre for Distance and Online Education

Third Semester

(Under CBCS)

M.Sc.-IT

Paper: INF 3066

DATA MINING AND WAREHOUSING

Contents: **Page No.**

Cover Page Designing:

Bhaskar Jyoti Goswami	GUCDOE
Nishanta Das	GUCDOE

ISBN: 978-81-982028-6-4

October, 2024

BLOCK- I

UNIT: 1
INTRODUCTION TO DATA MINING

Unit Structure:

1.1 Introduction

With the advent of new technologies in the field of data collection and storage, ample amount of data are being accumulated in different organizations and enterprises. Different types of data are accumulated such as business and scientific data, medical and personal data, audio and video data, geographical data and many more. The information and the knowledge that can be retrieved from this data can be used for many decision-making process. Data mining is a technique which can extract meaningful and useful data and also some existing pattern or association from large volumes of data. It uses different algorithms and techniques to explore and analyse new types of data as well as old types of data in new ways.

1.2 Unit Objectives

After going through this unit, you will able to:

- Understand the term Data Mining
- Understand the types of data that can undergo data mining
- Describe the task of Data Mining
- Understand the Knowledge Discovery in Database
- Understand the difference between Data Mining and Database System

1.3 Data Mining

Data mining is the process of discovering useful, understandable and hidden data in large data repositories. It uses techniques that can study the pattern of the data provided and predict the outcome of a future observation. It can analyse the data and find some hidden relationship that might exist in the vast amount of data such as in medical data. Patients suffering from a particular disease might have a relationship between their age and their lunch pattern which can only be discovered by using a Data Mining algorithm. Experts analysing databases may miss out some important information which might be beyond their expectation but using data mining techniques and algorithms it is easy to find such type of information. Data Mining techniques can be used to study sales data in a shopping mall by taking time as a measuring parameter, the authorities can collect useful information with which they are enable to take decisions such as to introduce new products in a particular time of the year or to offer discounts for some products. With the help of data mining techniques information can be collected from the data available, also patterns and rules can be discovered amidst the huge amount of data. By studying the pattern data, different problems related to delivery of a stable network can be solved. Different patterns of dataflow through the routers in a network in a particular time of a day can be studied and with the help of the patterns found, optimal ways can be found out for the routers to handle the traffic in the best possible way.

The different definitions that are used to describe Data Mining are given below:

- Data Mining is the non-trivial extraction of implicit, previously unknown and potentially useful information from the data.

  An example of "non-trivial extraction of implicit information" would be to find the correlation between the average age and average income of the employees in a company.
- Data mining is the search for the relationships and global patterns that exist in large databases but are hidden among vast amounts of data such as the relationship between patient data and their medical diagnosis.

  An example of the above definition is that Data mining can be used to find the characteristics or patterns that usually occurs in persons suffering from mild dementia and predict whether a person might have Alzheimer's disease in future. This can be done by analyzing the medical databases and find out a potential relationship that might exists in the data.
- Data mining refers to using of variety of techniques to identify nuggets of information or decision making knowledge in the database and extracting these in such a way that they can be put to use in areas such as decision support, prediction, forecasting and estimation.

  In any enterprise, a large amount of transactions are done in daily basis and which leads to accumulation of a massive volume of transaction data. These data with the help of Data mining techniques can be used for important decision making processes.
- Data mining discovers relationship that connects variables in a database which later can be used in some decision support system.
- Data mining is the process of discovering meaningful, new correlation patterns and trends by shifting through large amounts of data stored in repositories using pattern recognition techniques as well as statistical and mathematical techniques.

STOP TO CONSIDER

In 1989 Data Mining was first introduced by Gregory Piatetsky Shapiro, but by the name of Knowledge Discovery Databases (KDD). This term later found huge popularity in the field of AI and Machine Learning as Data Mining.

SAQ

Give an example how Data Mining can be used in social media?

1.3.1 Kinds of Data that can be mined

The different kinds of data that can be mined are:

- **Flat files:** Flat files are the simplest form of files which are either in text or binary format with a structure familiar to the data mining algorithm and hence data from it can be easily extracted. Flat files are usually represented by data dictionary such as CSV files.
- **Relational databases:** Relational database consists of data that is being organized in the form of tables with columns representing attributes and rows representing tuples. Each tuple in a table or relation is identified by a unique attribute which is called a key. The main characteristic of a relational database is to establish relationships between different tables that exist in a database using a primary key. Relational databases are managed by database management systems such as MySQL, Oracle, SQL Server, and PostgreSQL.
- **Data warehouse:**A data-warehouse is defined as the collection of data integrated from multiple sources that is used for solving queries and decision making. Enterprise data-warehouse, Data Mart and Virtual Warehouse are the three main types of data warehouse.
- **Transaction database**: A transaction database consists of a set of records where each record represents a transaction identified by a unique timestamp (transaction ID) and a set of items creating up the transaction.
- **Multimedia database:** Multimedia database consists of data that includes video, images, audio and text media. Object oriented databases are used for storing these kind of data. Due to the high dimensionality of multimedia databases, extraction of data becomes a challenging task. With the help of computer vision, computer graphics, image interpretation and NLP methodologies data mining is done in such type of database.
- **Spatial database:**Spatial databases store geographical information such as maps and global/regional positioning of

a place. It stores data in the form of coordinates, topology, lines, polygons, etc.

- **Time-series database:** A time-series database stores sequences of values or events accessed over repeated measurements of time (e.g., hourly, daily, weekly) such as stock exchange data, user logged activities and measurement of weather temperature, wind. These type of data requires real time analysis. These kind of data are indexed by time and date
- **World Wide Web:** This is the most heterogeneous and dynamic repository as it collects data from multiple sources identified by URL through the use of web browsers via the internet network. WWW comprise three main components: content, structure and usage of the web.

1.4 Knowledge Discovery Database

Knowledge Discovery Database (KDD) is an interdisciplinary area focussing upon technologies for extracting useful knowledge from data. KDD methodologies are the need of the hour due to the rapid growth of online data and also use immense databases. The various steps in the knowledge discovery process include data selection, data cleaning and processing, data transformation and reduction, data mining algorithm selection and finally the post processing and the interpretation of the discovered knowledge. Data mining is only one of the many steps involved in knowledge discovery in databases. It is the application of specific algorithms to extract the different patterns from data. Fayyad et al distinguish between KDD and data mining by giving the following definitions.

Knowledge Discovery in Databases is the process of identifying a valid, potentially useful and ultimately understandable structure in data. This process involves selecting or sampling data from a data warehouse, cleaning or preprocessing, transforming or reducing it (if needed), applying a data mining component to produce a structure, and then evaluating the derived structure.

Data Mining is a step in KDD process concerned with the algorithm means by which patterns or structures are enumerated from the data under acceptable computational efficiency limitations.

CHECK YOUR PROGRESS

1. List one difference between Data Mining and KDD

1.4.1 Stages of KDD

The different stage that comprises the process of KDD are given below:

- **Data Selection:** This step is concerned with the selection of the relevant data needed for executing a particular Data Mining task. While selecting data the subset of the data to be selected, the size of the dataset and the time period for which it is to be considered are decided in this phase.
- **Data Pre-processing:** This step involves cleaning of the data where unnecessary data is removed and strategies are designed for dealing with missing and corrupted data.
- **Data Transformation:** This stage involves the creation of appropriate data that will be suitable for the Data mining algorithm to process is done. This mainly involves dimension reduction as Data mining algorithms works better when the number of attributes of the data is lower or dimension is less. The benefits of dimensionality reduction are that it eliminates irrelevant features and reduce noise and can easily be visualized. Finally, the amount of time and memory required by the data mining algorithm is reduced with a reduction in dimensionality.
- **Data Mining:** This step evaluates the different data mining algorithms and selects the most appropriate algorithm suitable for the defined data mining task. The algorithm selected is then applied to the concerned data to find the implicit relationships and the different interesting patterns that exist in data.
- **Data Interpretation:** The interpretations obtained after conducting the Data Mining task are then applied to the application domain. The discovered patterns are used in decision making process.
- **Data Visualisation:** Finally Data Visualization helps the analyst to gain deeper understanding of the data and using in depth visualization, analyst can find out certain trends in data which if further explored can give very important piece of information. It helps the users to

examine large volumes of data and detect the patterns visually. Maps, charts and graphical representations allow data to be presented in such a manner that it becomes easy for the users to interpret the meaning from it.

The different phases of KDD is depicted in figure 1:

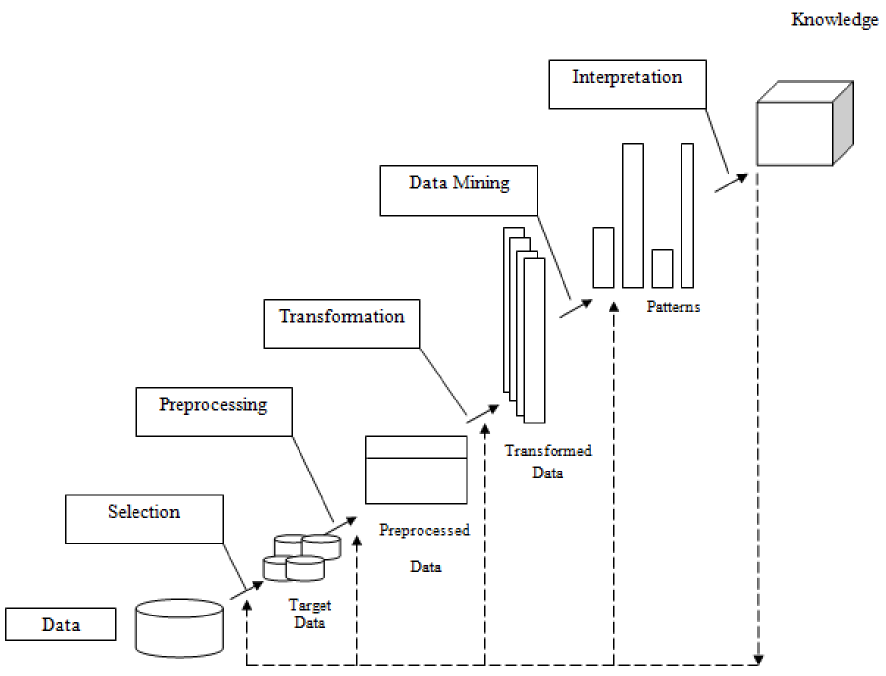


Fig 1: Phases of KDD

CHECK YOUR PROGRESS

2. What are the benefits of data transformation in KDD?

1.5 Data Mining Vs Database Systems

Database systems or Database Management Systems uses SQL to suffice the different queries and hence they are useful for query triggered data exploration. In case of Data mining, it supports automatic data exploration. If a user knows the data that needs to be retrieved then a DBMS query is sufficient. But if a user knows vaguely about the patterns and relationships that exist in the data then data mining techniques are useful. One of the important task of

Data mining is hypothesis testing which after formulation requires sifting through the database to test it. This can be handled by a DBMS query. So DBMS supports some of the primitive data mining tasks.

On the other hand a Data mining system may handle or use a relational database in two ways: either loosely coupled or tightly coupled or it may not use it at all.

Many data mining systems may use a database simply as a data repository from which it downloads data to it's own memory structures as it have it's own memory space. This helps in optimizing memory management specific to the data mining algorithm but these systems ignore the DBMS concepts of recovery and concurrency.

When Data Mining uses DBMS as loosely coupled then it uses DBMS only for storage and retrieval of data whereas as when DBMS is used as tightly coupled by Data Mining then it goes where the data naturally resides. In other words all data and processing is done at the database end without bringing data from the database to the data mining area. This is done to avoid performance degradation and the advantages of database technologies are fully utilized.

1.6 Summing Up

- Data mining is the process of discovering useful, understandable and hidden data in large data repositories.
- It can analyse the data and find some hidden relationship that might exist in the vast amount of data such as in medical data.
- The different data that can be mined are flat files, relational data, warehouse data, transaction data, multimedia data, spatial data, time series data as well as web data.
- Knowledge Discovery in Databases (KDD)is the process of identifying a valid, potentially useful and ultimately understandable structure in data.
- KDD involves selecting or sampling data from a data warehouse, cleaning or pre-processing, transforming or reducing it (if needed), applying a data mining

component to produce a structure, and then evaluating the derived structure.

- Data Mining is one of the steps in KDD.
- The different stages of the KDD process are data selection, data pre-processing, data transformation, data mining, data interpretation and data visualization.
- Database systems or Database Management Systems uses SQL to suffice the different queries and hence they are useful for query triggered data exploration.
- In case of Data mining, it supports automatic data exploration.

1.7 Answers to Check Your Progress

1) Knowledge Discovery in Databases is the process of identifying a valid, potentially useful and ultimately understandable structure in data. This process involves selecting or sampling data from a data warehouse, cleaning or pre-processing, transforming or reducing it (if needed), applying a data mining component to produce a structure, and then evaluating the derived structure.

Data Mining is a step in KDD process concerned with the algorithm means by which patterns or structures are enumerated from the data under acceptable computational efficiency limitations.

2) Data transformation mainly does the task of dimensionality reduction which helps in eliminating irrelevant features and reduces noise and can also be easily visualized. Finally, the amount of time and memory required by the data mining algorithm is reduced with a reduction in dimensionality.

1.8 Possible Questions

1) Who first introduced the concept of Data mining?

2) What is Data Mining?

3) What is the purpose of Data Mining?

4) How can Data mining be used for medical diagnosis?

5) What is a Data Warehouse?

6) Define transaction database.

7) How is KDD and Data Mining related?

8) Explain the different steps followed in KDD.

9) Mention the differences between Data Mining and Database Management System.

10) "Data Mining is the non–trivial extraction of implicit information". Elaborate.

1.9 References and Suggested Readings

1. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

2. Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

---×---

UNIT: 2
DATA MINING TECHNIQUES AND APPLICATIONS

Unit Structure:

2.1 Introduction

In the earlier chapter we have learnt about the concept of data mining along with the types of data that can be mined. In this chapter we are going to learn the different data mining techniques. The two fundamental goals of data mining are: prediction and description. The objective of prediction is to make use of existing variables in the database in order to predict unknown or future values of interest and description focuses on discovering patterns describing the data and the subsequent presentation for user interpretation. These two objectives are fulfilled by the different data mining techniques. We will also learn how data mining is applied in different areas like business, e-commerce, science, engineering and healthcare.

2.2 Unit Objectives

In this unit you will learn

- A brief explanation of all the data mining techniques like discovery of association rules, classification, clustering, genetic algorithms, support vector machines etc.
- Application of data mining in different areas like business, e-commerce, science, engineering and healthcare.

2.3 Data Mining Techniques

DM techniques can be classified into two types:

- User-guided or verification driven data mining
- Discovery driven or automatic discovery of rules

In most of the data mining techniques a combination of both the above two approaches are present.

2.3.1 Verification Model

The verification model takes a hypothesis formed by the user and test to verify its validity. The emphasis is on the user who is responsible for formulating the hypothesis and issuing the query on the data to affirm or negate the hypothesis. For example with a limited budget in mailing campaign while launching a new product, it is very important for a supermarket to identify the section of the customers who are most likely to buy the new product. Here the user has to formulate a hypothesis to identify the potential buyers and their common characteristics. This is done by querying the historical information about the different customers and their common characteristics while purchasing a particular product. By having idea about it, the hypothesis is refined repeatedly until the required limit is reached. A user may come up with a new hypothesis or refine the existing one and verify it against the database. A variety of techniques like queries, multidimensional analysis and visualization are used to guide the exploration of the data being inspected.

2.3.2 Discovery Model

Discovery model discovers important information automatically hidden in the data. The data is analysed and examined in search of frequently occurring patterns, trends and generalizations about the data without any intervention or guidance from the user. The manner in which the rules are being discovered depends upon the data mining application. Let us take the example that we have considered in the verification model i.e. a supermarket launching a new product. In discovery model, there is no formulation of any hypothesis by the user rather the system automatically groups the customers with common characteristics. The typical discovery driven tasks are:

- Discovery of association rules
- Discovery of classification rules
- Clustering
- Discovery of frequent episodes
- Deviation detection

Let's have a brief explanation of all the above discovery driven tasks

- Discovery of association rules: An association rule is an expression of the form $X \Rightarrow Y$, where X and Y are the sets of items and it interprets that a transaction of a database that contains X tends to contain Y. Given a database, the goal is to discover all the rules that have support and confidence greater than or equal to the minimum support and confidence respectively.
  Let $L = \{l_1, l_2, l_3....l_m\}$ be a set of literals called items. Let D, the database, be a set of transactions, where each transaction T is a set of items. T supports an item x if x is in T. T is said to support a subset of items X , if T supports each item x in X. $X \Rightarrow Y$ holds in confidence c, if c% of transactions in D that support X also support Y. The rule $X \Rightarrow Y$ has support s in transaction set D if s% of the transactions in D support $X \cup Y$. Support means how often X and Y occur together as a percentage of the total

transactions. Confidence measures how much a particular item is dependent on another.

- Discovery of classification rules: Classification rule involves in finding the rules that classifies the data into disjoint groups. It mainly assigns class labels to each of the class after analysing the characteristics and features of the data. The input to the classification model is the training dataset whose class labels are know and after the model learns the characteristics or the labels very well, it aims to assign a class label to the future unlabelled records. The classification model gains enough knowledge after going through the training dataset that it can predict the class label of an unlabelled record. A set of classification rules are generated by such a classification process, which can be used to classify future data and develop a better understanding of each class in the database. This type of learning is termed as supervised learning.

- Clustering: Clustering is the process of grouping data having similar characteristics i.e. similar trends and patterns into different groups. Clustering comprises a great number of data mining algorithms. For clustering the data into groups, a similarity metric must be available. There is another way to cluster groups based on building set functions that measures the particular property of a group. The objectives of clustering are:
    1. To uncover natural groupings
    2. To initiate hypothesis about the data
    3. To find consistent and valid organization of the data

- Discovery of frequent episodes: Frequent episodes are the sequences of event that occur frequently close to each other and are extracted from time sequences. To consider an episode as frequent, it's frequency must exceed some predefined threshold specified by the user and also it is domain dependent.

  If R is considered a set of event types, then an event is a pair (A, t) where $A \in R$ is an event type and t is an integer, then we can calculate the occurrence time of the event. An event sequence s of R is a triple (T_s, T_c, S) where $T_s < T_c$ are integers . T_s is the starting point and T_c is the ending time.

$S = \{(A_1,t_1), (A_2,t_2),(A_n,t_n)\}$ is the ordered sequence of events, such that $A_i \in R$ and $T_s \leq t_i \leq T_c$ for all i=1,2….n-1.These episodes can be of three types. The episodes that occur in sequence are called serial episodes. The episodes where there are no constraints on the order of the event types are called parallel episodes. Suppose A and B are two events given but the order in which these events occurred is not mentioned then bot the events are called parallel episodes. Then there is non-serial and non-parallel episodes which occur in a sequence if the occurrences of A and B precede an occurrence of C, and there is no constraint on the relative order of A and B given.
These type of event sequences can be used in alarm alarm sequences, cloud data, network data, stock data, malicious attacks, movements, and customer transactions

Deviation detection: Deviation detection is often a source of true discovery as it identifies outliers in a particular dataset and finds out the reason as to whether they are due to noise or other impurities being present in the data or due to trivial reasons. It is usually applied to database segmentation. It discovers new rules as the outliers express deviation from previously known expectation and norm. The deviations can be obtained by comparing the values of measures of current data and previous data as well as also normative data. Deviation detection is usually applied in forecasting, fraud detection, customer retention etc.

2.3.4 Neural Networks

Neural Networks are new paradigm in computing which involves developing mathematical structures with the ability to learn. As in human brain, neural network also consists of some interconnected cells that works as a set of processing elements called neurons. These processing can identify patterns in data as well as also derive meaning from complicated or imprecise data. The neural network first learns to extract the features of the data that is being fed into it and once it is trained, it is tested to predict the outcome of the unknown or unlabelled data. Neural network is mainly used for classification. This will be discussed in details in the later chapters.

2.2.4 Genetic Algorithms

Genetic Algorithm is based on the concept of Darwinian Theory “Survival of the fittest”. It works with a population of individual strings, each representing a possible solution to the problem considered. Each individual string is assigned a fitness value which is an assessment of how good a solution is, to a problem. The high fit individuals are selected for participating in the reproduction process by cross breeding with other individuals of the population. This yields to the production of new offspring which share some of the characteristics with each of it’s parents. The less fit individuals are not considered for the reproduction process and hence die out. A whole new population of possible solutions to the problem is generated by selecting the best individuals from the current generation which contain characteristics better than their ancestors. This process continues generations after generations thus resulting in a population containing the best characteristics which provides solutions that fits to the problem in hand. Thus we get a optimal solution to a problem.

2.3.5 Rough Set Techniques

A set is defined as a collection of distinct and well-defined objects. The elements of the set can be uniquely determined and these elements form the set. For example when we say ‘a set of even numbers’, we exactly know what the set may contain. But in many cases we may not be able to say very clearly about a set. Suppose we assume ‘a set of good boys’ then we cannot measure the goodness of a person. We can only define certain attributes which are most likely possessed by a good boy. These are called vague, imprecise or uncertain notations. Rough set is a tool for studying imprecision, vagueness and uncertainty in data. The concept of vagueness has been used in fields like Artificial Intelligence which uses reasoning based on vague notations.

2.3.6 Support Vector Machines

A Support Vector Machine (SVM) is a supervised learning machine learning algorithm that can be used to solve complex classification, regression and outlier detection problems. In the SVM algorithm, we try to create an optimum hyper plane that distinguishably segregates the data points of different classes based on predefined classes, labels, or outputs. SVM's are potentially designed for binary classification. However they are also used in a number of applications such as healthcare, natural language processing, signal processing applications, and speech and image recognition fields.

Check Your Progress

1. Mention difference between classification and clustering.
2. What concept is followed by Neural Networks?
3. Mention two applications of Deviation detection.
4. What is the theory behind Genetic Algorithm?
5. How will you define vagueness in context of Rough Set technique?
6. What is the purpose of SVM?

2.4 Data Mining Application Areas

The application of data mining comprises almost all the fields starting from business, e-commerce, science, medical and engineering. Let's have a look on the different applications of data mining:

Market Management: Data mining helps in developing the customer relationship management. Any business company targets the customers satisfaction which is only possible if customer service is properly implemented. Customer service helps in creating and representing any brand and it's image. Deep customer analysis with the help of data mining techniques can help in gaining information about the customer and also their journey in buying or declining a certain product. The source of these information are customer feedback, surveys, social media comments, business emails, and etc. Other application areas include cross selling which performs associations/ correlations between product sales and target marketing which clusters customers having similar characteristics such as spending on a particular product or income etc. Data mining also helps in creating a customer profile by which the seller can make strategies for launching new products based on the mass interest.

Risk Management: Risk management involves planning strategies by business enterprises to retain the existing customers so that they may not switch to their competitors. Data mining also helps in forecasting the outcome for an investment done on a particular aspect and is it profitable or may face lose.

Fraud detection: Data mining helps in detecting frauds in the field of telecommunication and also in bank service such as credit cards services. It takes into account the time of such fake calls, duration of the call, the destination of the call and also the day and the week. It analyses the pattern of such events and tries to find out the way in which it deviates from the normal pattern.

Healthcare: A huge collection of data is accumulated in different health care organizations like hospital, insurance companies and concerned government agencies. These data are analysed to understand the underlying relationship which further helps in determining the procedures and clinical protocols to be most effective so that it is possible to deliver best health care to maximum number of people.

Business and E-commerce data: Business transaction involves tracking million of transactions performed by the customers and detecting fraudulent transaction or predicting the customers which might most likely migrate. In case of E-commerce data is analysed to find the buying pattern of the customer, detecting risk pattern and also to meet the demands of online transactions.

Scientific and Engineering: Data mining is used also in discovering hidden pattern and relationship in scientific and engineering data. There is a wide variety of online databases containing information about disease, cellular function, drugs etc. Databases containing genomic sequencecan be used to identify gene expression patterns in genomic data or to uncover associations between risk factors and disease outcomes in epidemiological studies. By applying data mining techniques, scientists can analyse complex data sets and gain insights that may not be apparent using traditional statistical methods. There are large datasets consisting of terabyte to petabytes of sensor data produced from satellites, buoys, balloons and a variety of other sensors. A large amount of data is also accumulated from various instruments in the fields of astronomy, high energy physics and nuclear physics which can be analysed by data mining techniques to understand the causal relationships amongst this data. Simulations also produces ample amount of data as it is considered as a third mode of science supplementing theory and experiment. Data mining in this case provides a critical link between theory, simulation and experiment.

Multimedia data: Multimedia data are increasing day by day as people today are grateful for this technology to retrieve as well as to upload documents easily without consuming time. As a result archiving multimedia data such as documents, audio, video and images is becoming easier but it’s becoming harder to extract meaningful information from the archives as the volume grows larger. Hence data mining is the only way out to discover the unknown patterns and relationships present in it.

STOP TO CONSIDER

Data mining used in education sector is termed as Educational Data Mining method (EDM). It is used in educational task such as predicting student’s performance, student’s enrolment in higher education, teacher’s teaching performance, curriculum development and predicting student placement opportunities.

2.5 Summing Up

1. DM techniques can be classified into two types: User-guided or verification driven data mining and Discovery driven or automatic discovery of rules.

2. The verification model takes a hypothesis formed by the user and the emphasis is on the user for issuing the query on the data to affirm or negate the hypothesis.

3. Discovery model discovers important information automatically hidden in the data.

4. An association rule is an expression of the form X⇒Y, where X and Y are the sets of items and it interprets that a transaction of a database that contains X tends to contain Y.

5. Classification rule involves in finding the rules that classifies the data into disjoint groups. It mainly assigns class labels to each of the class after analysing the characteristics and features of the data.

6. Clustering is the process of grouping data having similar characteristics i.e. similar trends and patterns into different groups.

7. The objectives of clustering are:

To uncover natural groupings
To initiate hypothesis about the data
To find consistent and valid organization of the data

8. Frequent episodes are the sequences of event that occur frequently close to each other and are extracted from time sequences.

9. Deviation detection is often a source of true discovery as it identifies outliers in a particular dataset and finds out the reason as to whether they are due to noise or other impurities being present in the data or due to trivial reasons.

10. Neural Networks are new paradigm in computing which involves developing mathematical structures with the ability to learn.

11. Genetic Algorithm is based on the concept of Darwinian Theory "Survival of the fittest". It works with a population of individual strings, each representing a possible solution to the problem considered.

12. Rough set is a tool for studying imprecision, vagueness and uncertainty in data.

13. In the SVM algorithm, we try to create an optimum hyperplane that distinguishably segregates the data points of different classes based on predefined classes, labels, or outputs.

14. Data Mining application covers a huge area as it involves market management, risk management, fraud detection, healthcare, business, e-commerce, scientific and engineering data as well.

2.6 Answers to Check Your Progress

1. Classification is the assignment of an unlabelled data in the right category/class where the classes or categories are predefined according to the features they exhibit. Clustering on the other hand is grouping of similar data instances together.

2. The neural network first learns to extract the features of the data that is being fed into it and once it is trained, it is tested to predict the outcome of the unknown or unlabelled data. Neural network is mainly used for classification.

3. Deviation detection is usually applied in forecasting, fraud detection, customer retention.

4. Genetic algorithm follows the Darwin theory "Survival of the fittest".

5. Vagueness means not clear. In context of rough set technique, sometimes it may so happen that we might not be able to define clearly a set as we cannot measure certain qualities like intelligence. A person may be defined as intelligent but we cannot exactly figure out the measure of the intelligence level.

6. In the SVM algorithm, we try to create an optimum hyperplane that distinguishably segregates the data points of different classes based on predefined classes, labels, or outputs. SVM's are potentially designed for binary classification.

2.7 Possible Questions

1.What are discovery driven task in data mining? Explain.

2. What is SVM? Explain.

3. How data mining is used in market management? Explain

4. Explain how data mining can help in fraud detection.

5.How scientific and engineering data is used by data mining? Explain.

2.8 References and Suggested Readings

1. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

2. Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education ndia.

---×---

UNIT: 3
INTRODUCTION TO DATA WAREHOUSING

Unit Structure:

3.1 INTRODUCTION

In the earlier units, the concepts of data mining have been discussed. In this unit, we are going to learn about the basic concepts of data warehouse. The concept of data warehouse was first introduced by Barry Devlin and Paul Murphy In 1980. They had proposed the movement of information from operational databases to decision support systems. A decision support system can provide great help to make strategic decisions from different information. In 1990, growth of different businesses was rapidly increased and as a result the complexity and competition in business were also increased. In that situation, the business organizations were required to make different strategic decisions which can help to reduce business complexities and increase the growth of their businesses. As a result, in 1990, the concept of data warehouse was emerged and the organizations had initiated to construct data warehouses.

3.2 OBJECTIVES

After reading this unit, learners are expected to be able to know:

- What is data warehouse?
- About the characteristics of data warehouse and why do we require it.
- About the approaches to build a data warehouse.
- About the differences between data warehouse and data base system.
- About the differences between data warehouse and data mining.

3.3 WHAT IS DATA WAREHOUSE?

According to W.H. Inmon, data warehouse is a subject-oriented, integrated, time varying, non-volatile group of data that support the decision making process of an organization. So, a data warehouse is actually a storehouse of integrated operational data which can help a decision support system to provide solutions to complex analytical questions. A decision support system also provides answers to the structured and ad hoc queries with the help of data warehouse.Data warehouse provides a platform to organize and maintain very large amount of data and deliver data analysis according to the requirements.Data are regularly extracted subject wise from different types of sources and loaded into data warehouse using different loading approaches. In data warehouse, it is very important to maintain a common data format for all data as it will be used in decision support systems and data inconsistency must be removed from data warehouse. For this purpose, data cleaning, data transformation and data integration techniques are applied. Data cleaning techniques are used to remove available errors from data in data warehouse. In general, these techniques consist of transformation rules, domain-specific knowledge, parsing and fuzzy matching and auditing. Data transformation techniques are used to convert data with diverse structures to data with a uniform format.

It has been observed that gradually data warehouse becomes very important in every domain and nearly every big organizations and business segments are starting to use it. It plays a crucial role for the business organizations in the process of developing effective business policies for the growth of related businesses. At the time of inception, data warehouse was primarily utilized to prepare reports and to reply already existing queries. Later, it was used to analyse complete and summarized data. Then in the next development, data warehouse was used to perform strategic operations efficiently. In the latest development, it becomes an effective tool for knowledge discovery mechanism and decision making system of an organization.

3.4 CHARACTERISTICS OF DATA WAREHOUSE

We have already discussed about the meaning of data warehouse and in this section, the important characteristics of data warehouse are presented in the following points.

- Data warehouse is responsible for storing large amount of recent and long duration historical data so that it can provide sufficient information to the decision making system of an organization for delivering efficient strategic decisions in the process of continuous development of the organization.
- Data available in data warehouse are subject-oriented and these are extracted regularly from different types of sources or applications. It means data are organized based on different subjects in data warehouse. For example data warehouse of a banking system stores subject wise data like customer record, employee record, transaction record etc.
- Data warehouse can also be referred as a group of integrated data because all necessary data for decision support system are extracted from different heterogeneous sources and then they are integrated together so that it can

be stored in data warehouse. The formats of these extracted data are different depending on their sources. But these integrated data must be stored in data warehouse with a common data format so that it can be used efficiently and easily for decision support process. So data cleaning and data integration techniques are applied to eliminate the variations from different source data so that all data in a data warehouse are available with a common data format.

- Separate storage devices are used for data warehouse to store data and as a result transaction processing, recovery and concurrency control mechanism are not required in data warehouse.
- In general, data are not allowed to be changed or modified or updated in data warehouse. Only necessary new data are regularly extracted to the data warehouse. So data warehouse can be termed as a non-volatile group of data. But in some situations, data in a data warehouse are allowed to be changed or modified or updated when the corresponding data at the source are updated. In such cases, data warehouse is updated or refreshed periodically as designed by data administrator.
- Data warehouse stores data according to different time periods to achieve greater accuracy in data analysis. So, data warehouse can also be termed as a collection of time-varying data. For example, a data warehouse of a banking organization may stores year wise customer records.
- Different types of data warehouse users perform their jobs with the help of data warehouse. But in general a data warehouse supports a small number of users.

3.5 BENEFITS OF DATA WAREHOUSE

It has been witnessed that in recent times, different organizations are continuously benefited in different ways by developing data warehouses. Some of these benefits of data warehouse are stated in the following points.

- Better business strategies can be delivered with the help of the strategic information provided by data warehouses.

- We already know that the data format for all available integrated data in a data warehouse is uniform. As a result, the data analysis and sharing of analyzed information becomes simpler with the help of data warehouse. The cost and time required for data analysis is also decreased considerably.
- Data warehousing can be able to reduce the risk of error in data interpretation and increase data reliability.
- Access of necessary data from data warehouse is simpler as data are stored separately in single location.
- Data security can also be maintained in data warehouse by providing protected data access. In this mechanism, only authenticated users can access those data that are specific to them.

3.6 APPLICATIONS OF DATA WAREHOUSE

In general, data warehouse can be used in the following operations.

- Discovery of correct information
- Providing necessary information
- Identification of hidden patterns in available data
- Identification of relationships amongst data
- Hypothesis testing
- Perform data analysis and sharing the analysis

Applications of data warehouses are witnessed in different areas. In general, data warehouses have been continuously used in the following areas.

- Banking and financial sector
- Educational data analysis for educational institutions
- Biological data analysis
- Telecom industry
- Management of logistics
- Management of inventory
- E-commerce sector
- Decision support mechanism of business organizations
- Customer data analysis
- Travel and Tourism industry

- Insurance organizations
- Maintenance of product quality

STOP TO CONSIDER

Construction and utilization of a data warehouse can be referred as data warehousing.

3.7 DATA WAREHOUSE USERS

Data warehouses are used by different types of users to fulfill their purpose. End users like sales representatives can use data warehouse for particular information by accessing reports which are already exists. In some cases, limited access to the data warehouse for related information is also provided to the customers of an organization. Business analyst analyse data warehouse information to discover current trends and patterns for an organization so that decision makers can take proper strategic decision to increase the growth of their businesses. Business analysts are capable of developing ad-hoc queries to data warehouse so that they can acquire required information from data warehouse. Data scientists also analyse data warehouse information but they perform it in more advanced ways by using different techniques like pattern recognition, machine learning techniques etc. They must have complete knowledge and access to the data warehouse. They can provide models for future development by analysing data warehouse information of an organization. Data administrators are responsible for designing and managing the data warehouse. IT administrators are required for the maintenance and security of data warehouse. They are responsible for providing technical supports to solve technical problems.

Broadly data warehouse users can be categorized into three classes based on their ability and level of interaction with the warehouse. These three classes are Casual user, Power user and Expert user.

- **Casual user:** Casual users have limited access to data warehouse. They can access only predefined reports and execute only predefined queries in data warehouse. For example: End users.

- **Power user:** Power users are capable of producing simple ad hoc queries to data warehouse and also able to execute them. They can access the outcome of simple queries and reports. For example: Business analysts. Power users require access tools to use data warehouse for their purpose.
- **Expert user:** Expert users can create complex queries for data warehouse information. They can perform data analysis to discover new information from data warehouse. Expert users have complete knowledge about the data warehouse and required tools for their purpose. For example: Data scientists.

3.8 CONSTRUCTION OF DATA WAREHOUSE

In general, construction of a data warehouse can be performed in two ways that are top-down approach and bottom-up approach.

3.8.1 Top-down Approach

Top-down approach to construct a data warehouse was proposed by Bill Inmon. In this approach, initially a centralized repository is constructed based on an enterprise data model. This centralized repository is referred as Enterprise Data Warehouse (EDW). In the next step, business necessities are identified based on enterprises. Then according to the requirements, different data marts based on different subject areas are constructed to build the data warehouse. The advantage of this approach is that a centralized repository is available in the data warehouse to store consistent normalized data and new data mart can be built without difficulty from the data warehouse depending upon the requirements. Top-down approach is useful when different subject specific requirements of the data warehouse can be clearly identified and these requirements will not be changed in future. Otherwise this approach may not be efficient enough to build a data warehouse. Top-down approach requires more time to build a data warehouse and it is a costly approach.

3.8.2 Bottom-up Approach

Bottom-up approach to build a data warehouse was introduced by Ralph Kimball. In this approach, initially, different data marts specific to different subject areas and business requirements are created independently at different times. In the next step, data marts are united together to construct the desired data warehouse. The advantage of this approach is that the process to construct a data warehouse can be started without waiting for all requirement details of the data warehouse and as a result, a lesser amount of time is needed for the initial set up of the data warehouse. In this approach, new subject area can be easily included to the data warehouse by creating and integrating a new data mart to the data warehouse after collecting necessary requirement details specific to the subject area.The disadvantage of this approach is that the process to integrate a new data mart to the data warehouse is complex.

STOP TO CONSIDER

A data mart is a storehouse which is smaller in size than data warehouse. It collects data that are associated to a specific subject area or department within an organization. Data mart is constructed to fulfil the requirements of specific group of users and departments of an organization.

3.8.3 Data Warehouse Designing

The most important step to build a data warehouse is its designing. Now, at first, all constituents of the data warehouse, all kind of sources for necessary data and types of all required information must be identified to design an effective data warehouse. Data warehouse designer have to communicate with the end users to identify their information requirements and it will help in designing a useful data warehouse. Selection of appropriate data integration techniques is also an important part of data warehouse design. Data warehouse designer are also responsible for selection of appropriate tools that will be used to implement a data warehouse for a particular organization. A data warehouse design must include proper methods and technologies to manage metadata repository. The process of data warehouse designing also includes designing appropriate data placement and data distribution policies.

3.8.4 Technical Requirements to Construct Data Warehouse

The technical requirements to construct a data warehouse are presented in the following points.

- A compatible hardware platform and related software tools are required to develop an efficient data warehouse.
- A data warehouse server must be designed and developed which can provide all services of data warehouse as required by different users. The server must have sufficient physical storage to accommodate large amount of necessary data for decision support processes. So, a compatible Database Management System will also be required to handle large amount of data and process complex ad hoc queries.
- Skilled data warehouse users are required for efficient technical support to handle different technical issues related to the data warehouse.
- A data warehouse must provide a repository for metadata. For this purpose compatible software and hardware are required in the development of a data warehouse.
- Compatible Database Management System is required to construct a data warehouse which can handle large amount of data
- Efficient communication networks are required to transfer large amount of data to the data warehouse from different sources. Most modern hardware technologies and compatible software should be used to configure the communication networks.

3.8.5 Basic Steps to Implement Data Warehouse

To implement a data warehouse, all types of requirements according to the data warehouse design are acquired and all related parts of the data warehouse are joined. In general, following basic steps are followed to implement a data warehouse.

- To initiate the implementation of a data warehouse, at first, all categories of information related to the data warehouse are collected. Then an efficient data warehouse design is prepared by analyzing this information and identifying different requirements of the warehouse.
- A data model is designed for the data warehouse based on the data warehouse design.
- Different sources of necessary data for the data warehouse are recognized and compatible Database Management System is selected to develop the data warehouse server. In this process, an efficient hardware platform has to be developed by selecting hardware devices that are capable enough to handle large amount of data.
- Efficient data collection operation is performed to extract necessary data from different heterogeneous sources. Appropriate data transformation, data cleaning and data integration techniques are applied to remove variations from extracted data so that these consistent data can be stored into the data warehouse with a uniform data format.
- Metadata repository is also constructed in the data warehouse so that Meta data can be properly preserved. The metadata should be accessible for all types of data warehouse users.
- Compatible software tools are selected so that the data warehouse can provide efficient services to its users.
- Maintenance of the data warehouse must be performed properly by skilled professionals and according to the requirement it should be updated.

3.8.6 Cost to Build a Data Warehouse

When an organization is going to develop a data warehouse then the cost for this purpose has to be estimated. The cost to develop a data warehouse may be different for different organizations depending upon their business requirements. Accurate estimation of this cost for an organization is not possible. The cost of data warehousing can be divided into three major parts that are Hardware cost, Software cost and Human resource cost.

Hardware Cost: Construction of a data warehouse requires various hardware units like CPU, storage devices, hardware platform to establish computer networks etc. So, at first a well-suited hardware platform is essential to construct a data warehouse that can fulfill the requirements of an organization. As a result, the cost to establish a hardware platform to construct a data warehouse has to be estimated. The hardware cost is depended on the amount of data, usage of data and number of users in a data warehouse. Increase in amount of data, usage of data and number of data warehouse users will raise the hardware cost. Additionally, to enhance the performance of a data warehouse, different hardware with most modern technologies has to be acquired and accordingly it increases the hardware cost.

Software cost: We all know that hardware requires appropriate software to perform its job efficiently. Compatible software or software tools are required for different hardware devices associated in data warehousing. As a result, cost has to be estimated to acquire the required software while constructing data warehousing. This software cost can be reduced by applying available compatible open source software. Conversely, requirement of software maintenance may increase the software cost in building data warehousing.

Human resource cost: Finally, skilled human resource is required to handle and maintain data warehouse. For example, skilled data warehouse users are required to regularly monitor and update the data warehouse. Data warehouse database is managed by different types of users like data administrators, software engineers, backend developers etc. So, the construction of data warehouse must include the cost to recruit required human resource. Every organization must provide trainings to their employees so that they can perform better in their jobs with the help of latest technologies. So, an additional cost is also associated with the construction of data warehouse to provide regular professional trainings for the human resource.

3.9 DIFFERENCE BETWEEN DATA WAREHOUSE AND DATA BASE SYSTEM

We have already learnt that database system is a type of software which provides various services like storing information, manipulation of data and data validation in a database. It offers secured and efficient access of information from a database. So, it can be clearly stated that database system is different from data warehouse. The differences between data warehouse and database system can be stated by the following points.

- We have already learnt that data warehouse contains subject based long duration historical data that are extracted from various types of sources. On the other hand, database system deals with application oriented short duration current data.
- Separate storage devices are used in data warehouse to store data as in general, data are not changed or modified in data warehouse. As a result, data recovery and concurrency control mechanism are not required in data warehouse. On the other hand data recovery and concurrency control are two most important services provided by database systems to handle issues that are occurred due to frequent data modifications by transaction processing functions.
- Data warehouse provide support to analyze data so that decision support system can deliver strategic decisions. On the other hand database system stores transactional data and it provide different services like data processing, data manipulation, data validation, data recovery etc.
- The main purpose of data warehouse is to provide efficient support to the decision support system so that it can deliver effective strategic decisions. But database system is primarily responsible for storing information, manipulation of data and data validation in a database and it can offer a minimal support in decision making process.

3.10 DATA WAREHOUSE VERSUS DATA MINING

The concept of data mining is already discussed in the earlier units and in this unit we have learnt about the basic concepts of data warehouse. It is observed that both these concepts are connected when data analysis is required to be performed for the decision making process of an organization. Both concepts are very crucial for an organization to take strategic decisions. It can be stated that knowledge or hidden information discovery can be performed by applying data mining process on data warehouse.

The differences between data warehouse and data mining are discussed in the following points.

- Data warehouse is a centralized storehouse for large amount of integrated subject wise historical data. But data mining is a collection of different techniques like pattern recognition techniques, machine learning techniques etc.
- Data warehouse mainly provides support to decision making systems for efficiently delivering strategic decisions. On the other hand data mining process is applied to perform data analysis and discover hidden patterns and relationships so that data classification and prediction of future development can be achieved.
- The processes involved in a data warehouse are data extraction, data cleaning, data transformation, loading, updating the warehouse, querying and reporting.On the other hand, the processes involved in data mining are selection of suitable techniques or algorithms, preparation of training models, evaluation of acquired information and providing hidden knowledge or information.

CHECK YOUR PROGRESS

1. (i) Find out the false one from the following statements.

A. Data warehouse stores integrated data.

B. Data warehouse stores only short duration transactional data.

C. Data warehouse helps in data analysis for decision support mechanism.

D. None of the above.

(ii) Data warehouse stores ______ data.

A. subject-oriented

B. volatile
C. transactional
D. None of the above.

(iii) Find out the correct one from the following statements.
A. Recovery and concurrency control are two important services provided by data warehouse.
B. Data security cannot be maintained in data warehouse.
C. Data warehouse can help to discover correct information.
D. None of the above.

(iv) Which of the following operation don't require data warehouse?
A. Data analysis
B. Identification of hidden patterns
C. Hypothesis testing
D. Data processing

(v) Data scientist is an ______ of data warehouse?
A. Casual user
B. Power user
C. Expert user
D. None of the above

(vi) Find out the correct statement for Top-down approach to build data warehouse.
A. Initially, a centralized repository is constructed based on an enterprise data model.
B. Initially, different data marts specific to different subject areas and business requirements are created independently at different times.
C. The process to construct a data warehouse can be started without waiting for all requirement details of the data warehouse.
D. All of the above.

(vii) Find out the correct statement for Bottom-up approach to build data warehouse.
A. Initially, a centralized repository is constructed based on an enterprise data model.

B. Initially, different data marts specific to different subject areas and business requirements are created independently at different times.

C. The process to construct a data warehouse can be started without waiting for all requirement details of the data warehouse.

D. Both B and C

2. State whether the following statements are true or false
   (i) Data warehouse organizes data according to different time periods to achieve greater accuracy in data analysis.
   (ii) Integrated data must be stored in data warehouse with a common data format.
   (iii) Business analysts are casual users of data warehouse.
   (iv) Top-down approach requires very less time to build a data warehouse.
   (v) Data marts are more flexible than data warehouse.
   (vi) The concepts of data mining and data warehousing are similar.

3.11 SUMMINGUP

- A data warehouse is collection of integrated operational data which can help a decision support system to provide solutions to complex analytical questions. All necessary data for decision support system are extracted from different heterogeneous sources and then they are integrated together so that it can be stored in data warehouse.
- Data warehouse stores and maintains very large amount recent and long duration historical data and deliver data analysis according to the requirements. In general, data are not allowed to be changed or modified or updated in data warehouse.
- Subject-oriented data are stored in data warehouse.
- The data format of all available data in a data warehouse is uniform. As a result, the data analysis and sharing of analyzed information becomes simpler with the help of data warehouse. The cost and time required for data analysis is also decreased considerably.

- Separate storage space is used for data warehouse to store data and as a result transaction processing, recovery and concurrency control mechanism are not required in data warehouse.
- Data warehouse stores data according to different time periods to provide greater accuracy in data analysis.
- Data warehouse provide strategic information so that better business strategies can be delivered.
- Data warehousing helps to decrease the probability of error in data interpretation and increase data reliability.
- Data security can also be maintained in data warehouse by providing protected data access to only authenticated users.
- The concept of data warehouse can be applied in the following operations.
    - Discovery of correct information
    - Providing necessary information
    - Identification of hidden patterns in available data
    - Identification of relationships amongst data
    - Hypothesis testing
    - Perform data analysis and sharing the analysis
- Data warehouses have been continuously used in different areas like Banking and financial sector, Biological data analysis, Telecom industry, E-commerce sector etc.
- Some examples of data warehouse users are End users, Business analyst, Data scientists etc.
- Broadly data warehouse users can be divided into three classes based on their ability and level of interaction with the data warehouse. These three classes are Casual user, Power user and Expert user.
- In general, top-down approach and bottom-up approach are two approaches used to construct a data warehouse.
- In top-down approach, initially a centralized repository is constructed based on an enterprise data model. Then business necessities are identified based on enterprises. Finally, according to the requirements, different data marts based on different subject areas are constructed to build the data warehouse.
- In bottom-up approach, initially, different data marts specific to different subject areas and business requirements are

created independently at different times. Then data marts are united together to construct the desired data warehouse.

- The first step to construct a data warehouse is data warehouse designing. All constituents of the data warehouse, all kind of sources for necessary data and types of all required information must be identified to design an effective data warehouse.
- All technical requirements must be fulfilled to construct an effective data warehouse.
- To implement a data warehouse, all types of requirements according to the data warehouse design are acquired and all related parts of the data warehouse are joined.
- The cost to construct a data warehouse can be divided into three major parts that are Hardware cost, Software cost and Human resource cost.
- Database system is different from data warehouse as it deals with application oriented short duration current data. Additionally, database system requires data recovery and concurrency control mechanism and it provide different services like data processing, data manipulation, data validation, data recovery etc.
- The concept of data mining is different from the concept of data warehouse as it is a collection of different techniques like pattern recognition techniques, machine learning techniques etc. Data mining can be applied on data warehouse to perform data analysis and discover hidden patterns and relationships so that data classification and prediction of future development can be achieved.

3.12 ANSWERS TO CHECK YOUR PROGRESS

1. (i) B ,(ii) A ,(iii) C ,(iv) D,(v) C,(vi) A , (vii) D
2. (i) True ,(ii) True ,(iii) False ,(iv) False ,(v) True ,(vi) False

3.13 POSSIBLEQUESTIONS

1. Define data warehouse. Write down the characteristics of data warehouse.
2. Write down the importance of data warehouse.
3. Give examples of data warehouse applications.

4. Explain the basic steps to construct a data warehouse.
5. Write a short note on the approaches to build a data warehouse.
6. How database system is different from data warehouse?
7. Explain different types of cost associated to the construction of data warehouse.
8. How data mining is related to data warehousing?
9. Write down the differences between data warehouse and data mining.

3.14 REFERENCES AND SUGGESTED READINGS

1. Berson, A., & Smith, S. J. (1997). Data warehousing, data mining, and OLAP. McGraw-Hill, Inc..
2. Inmon, W. H. (2005). *Building the data warehouse*. John wiley & sons.
3. Ponniah, P. (2004). *Data warehousing fundamentals: a comprehensive guide for IT professionals*. John Wiley & Sons.
4. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

---×---

UNIT: 4
MULTIDIMENSIONAL DATA MODEL AND OLAP OPERATIONS

Unit Structure:

4.1 Introduction

In today's data-driven world, efficiently organizing and analyzing large volumes of data is crucial for businesses. The Multidimensional Data Model offers a powerful framework for managing and interpreting complex datasets. This model, widely used in data warehousing and Online Analytical Processing (OLAP), allows users to examine data from various perspectives, facilitating insightful analysis and decision-making. Key components of this model include data cubes, dimensions, and measures, which together enable a more intuitive and flexible approach to data exploration. This unit will introduce you to the fundamental concepts, features, and operations associated with the Multidimensional Data Model.

4.2 Unit Objectives

After going through this unit you will be able to:

- Understand the fundamental concepts of the Multidimensional Data Model.
- Learn the stages involved in building a Multidimensional Data Model.

- Identify and describe the features of a Multidimensional Data Model.
- Comprehend the principles of dimension modeling.
- Gain insight into OLAP and its operations for efficient data analysis.

4.3 Multidimensional Data Model

The Multi-Dimensional Data Model is a sophisticated approach to organizing and structuring data in a database, providing a systematic and efficient way to manage information. Unlike traditional relational databases that primarily rely on queries for data access, this model offers a unique capability for users to analyses data by posing analytical questions related to market or business trends.

In contrast to relational databases, where users retrieve data through queries, the Multi-Dimensional Data Model empowers customers to interrogate data in a more intuitive and analytical manner. It enables users to swiftly obtain answers to their inquiries by efficiently creating and examining data.

This model finds prominent application in Online Analytical Processing (OLAP) and data warehousing. In the realm of OLAP, it serves as the foundation for systems designed to facilitate online analytical processing, allowing users to explore and understand data from multiple dimensions.
A key visual representation of the Multi-Dimensional Data Model is the use of data cubes. These cubes offer a dynamic way to model and view data, showcasing it from various dimensions and perspectives. The model is defined by dimensions, representing different categorical attributes, and facts, which are numerical measures associated with these dimensions. The arrangement of these dimensions and facts is often encapsulated in a fact table, providing a comprehensive structure for data representation.
The Multi-Dimensional Data Model is a powerful tool that not only organizes data within a database but also revolutionizes the way users interact with and analyze information. It is a fundamental part in OLAP and data warehousing, enabling a more insightful and rapid exploration of data in the complex landscape of business and market trends.

4.3.1 Working on a Multidimensional Data Model:

Building a Multidimensional Data Model involves several key stages, as outlined in the process you've described. Let's look into each stage for a more detailed understanding:

Stage 1: Assembling data from the client
The main objective of the first stage is to collect accurate and relevant data from the client. This stage comprises of communication with clients to understand the scope of data needed. This is done to provide clarity to clients on the data that can be obtained using the chosen technology. This also helps in collecting comprehensive data details.

Stage 2: Grouping different segments of the system
The second stage is to recognize and classify data into respective sections to make data application step-by-step. This includes the identification of different segments of the system and organizing data into logical groups for ease of application.

Stage 3: Noticing the different proportions
The third stage is to identify main factors or dimensions based on the user's perspective. Recognizing critical factors or dimensions that drive the design to understand user viewpoints and needs are the activities in this stage.

Stage 4: Preparing the actual-time factors and their respective qualities
The next stage uses identified factors to determine relevant qualities or attributes. This is achieved by associating the attributes with identified dimensions. This in turn helps in defining the characteristics or qualities associated with each dimension.

Stage 5: Finding the actuality of factors and their qualities
After the preparation of the actual time-factors, the separation of the actual data from collected factors is done by identifying and extracting real, meaningful data from the previously collected factors. This helps you to understand the significance of each factor in the model.

Stage 6: Building the Schema to place the data

The last stage is to create a schema to organize and structure the data based on the information collected. All the gathered data is used to create a schema and define the relationships between different dimensions and their attributes. In the end, you can develop a structured framework for organizing and storing data.

By following these stages, you are systematically progressing through the process of building a Multidimensional Data Model. This structured approach helps ensure that the model is based on accurate, relevant data and is organized in a way that aligns with the user's analytical needs. Each stage contributes to the overall effectiveness and efficiency of the data model.

4.3.2 Features of multidimensional data models:

Measures:

Measures are numerical values representing quantifiable data, such as sales, revenue, quantity, or any other metric that can be analyzed and compared. In a multidimensional data model, measures are typically stored in fact tables, providing the quantitative basis for analysis.

Dimensions:

Dimensions are attributes that provide context to measures. They describe the characteristics of the data being analyzed, such as time, location, product, or customer. Dimensions are stored in dimension tables and serve as the basis for organizing and categorizing the data.

Cubes:

Cubes are structures that visually represent the multidimensional relationships between measures and dimensions. They provide an intuitive way to view and analyze complex datasets. Each cell within a cube represents a specific intersection of dimensions, making it easier for users to understand the relationships between different aspects of the data.

Aggregation:

Aggregation is the process of summarizing data across dimensions and levels of detail. It allows users to view data at different levels of granularity. For example, data can be aggregated from daily sales to monthly or yearly totals, providing a broader perspective for analysis.

Hierarchies:
Hierarchies organize dimensions into levels of detail, facilitating navigation and analysis. For instance, a time dimension hierarchy might include levels such as years, quarters, months, and days. Hierarchies enable users to navigate data more efficiently and perform drill-down and roll-up operations.

OLAP (Online Analytical Processing):
OLAP is a category of multidimensional data models designed for efficient querying and analysis of large datasets. OLAP systems provide fast response times for complex queries, allowing users to interactively explore and analyze data. OLAP databases are optimized for analytical tasks, making them well-suited for decision support and business intelligence applications.

4.3.3 Advantages of Multi Dimensional Data Model

The following are the advantages of a multi-dimensional data model:

- Efficient Data Analysis:
    - Users can quickly analyze and gain insights from large datasets.
    - The model's structure allows for efficient data retrieval and analysis.

  Intuitive Representation:
    - Data is represented in a way that mirrors real-world relationships.
    - Cubes and dimensions provide an intuitive and visually appealing representation.

  Flexibility in Analysis:
    - Users can analyze data from various perspectives.
    - The model allows analysis based on different dimensions, offering flexibility.

  Fast Query Performance:
    - Quick response times for queries.
    - Pre-aggregated data and efficient storage structures contribute to fast query performance.

  Support for Complex Analysis:
    - Handles complex analytical queries effectively.
    - Supports operations like drill-down, roll-up, and slicing for in-depth analyses.

Hierarchical Organization:

- Organizes data into hierarchical structures.
- Hierarchies provide a structured way to navigate dimensions, aiding in data exploration.

Enhanced Decision Support:

- Facilitates decision-making processes.
- Offers a comprehensive view of data for informed business decisions.

Optimized for OLAP Systems:

- Tailored for Online Analytical Processing (OLAP).
- Provides a user-friendly environment for interactive and exploratory analysis.

These advantages collectively make the Multi-Dimensional Data Model a powerful tool for businesses and organizations seeking efficient and insightful data analysis capabilities.

4.3.4 Disadvantages of Multi Dimensional Data Model

The following are the disadvantages of a Multi Dimensional Data Model:

- Complexity in Design:
  - Design and implementation can be complex, requiring skilled professionals.

  Data Redundancy:
  - Potential redundancy in storing data across different dimensions.

  Limited to Specific Use Cases:
  - Suited for analytics but may not handle real-time updates or transactions efficiently.

  Scalability Challenges:
  - Scaling for large datasets may pose challenges.

  Difficulties in Real-Time Updates:
  - Challenges in accommodating continuous, real-time data updates.

  Steep Learning Curve:
  - Understanding and working with multidimensional data concepts may be challenging.

  Not Always Suitable for All Data Types:
  - May not be optimal for certain types of unstructured or semi-structured data.

  Cost of Implementation:
  - Implementation, especially at scale, can be expensive.

Dependency on OLAP Tools:

- Effectiveness often relies on specific OLAP tools, limiting flexibility.

Data Consistency Challenges:

- Ensuring consistency across dimensions and hierarchies can be challenging.

Check Your Progress:

1. Define the Multi-Dimensional Data Model. How does it differ from traditional relational databases?
2. Describe the key components of the Multidimensional Data Model.
3. What are the key features of a Multidimensional Data Model?
4. Explain the concept of OLAP and its importance in data analysis using the Multidimensional Data Model.

4.4 Data cube

A data cube is a concept within the field of multidimensional databases and OLAP (Online Analytical Processing) systems. A data cube allows data to be modeled and viewed in multiple dimensions. It is a multi-dimensional extension of two-dimensional tables of a relational database. It represents a multi-dimensional array of data, allowing users to analyze and explore information from different perspectives. Dimensions are the perspectives or entities with respect to which an organization wants to keep records. Each dimension may have a table associated with it, called a dimension table, which further describes the dimension. Dimension tables can be specified by users or experts, or automatically generated and adjusted based on data distributions.

Following are some components of a data cube:

1. Dimensions: A data cube is organized around dimensions, which represent the different facets or categories of data. Common examples include time, geography, product, and customer.
2. Measures: Measures are the numerical data points or metrics that users want to analyze. Examples include sales revenue, quantity sold, or any other measurable metric.
3. Hierarchies: Dimensions can be organized into hierarchies, representing levels of detail within a dimension. For example, a time dimension hierarchy might include levels such as year, quarter, month, and day.
4. Cells: Each cell in a data cube represents a unique combination of dimension members and holds the measure or metric value associated with that combination.

Structure of a Data Cube

1. Cuboids: Each level of abstraction in a data cube is called a cuboid. The base cuboid holds the lowest level of detail, and higher-level cuboids are formed by aggregating data.
2. Cells: Each cell in the data cube stores a measure value. The position of the cell is determined by its dimensions.

Consider a sales database where we want to analyze data based on three dimensions: Time, Geography, and Product.

Time	Location	Product	Sales
Q1	North	Widget	1000
Q2	South	Widget	1500
Q3	North	Gadget	2000
Q4	South	Gadget	2500

This table can be visualized as a 3D data cube with Time, Location, and Product as the dimensions and Sales as the measure.

Here are the key characteristics of a data cube:

1. Fast Query Performance: Data cubes are designed for fast query performance. Aggregated data is pre-computed and stored, enabling quick retrieval of summarized information.
2. Business Intelligence and Decision Support: Data cubes are a fundamental component of OLAP systems, providing a powerful tool for business intelligence and decision support. They enable users to interactively analyze large datasets to derive insights and make informed decisions.
3. Implementation in OLAP Systems: OLAP systems use data cubes as their underlying structure, and various OLAP tools provide interfaces for users to interact with and analyze data in a multidimensional space.

4.5 Dimensional modeling:

Dimensional modeling within a data cube is a structured approach to organizing data for efficient analysis in a multidimensional environment. It involves several key steps to create a model that enhances the understanding and exploration of complex datasets.

Firstly, dimensions are identified, representing categorizations such as time, geography, or product. These dimensions are organized hierarchically, creating a logical framework for data analysis. For instance, a time dimension might have hierarchies like year, quarter, month, and day.

Attributes are then defined within each dimension, providing additional details for analysis. For example, a product dimension may include attributes like product name, category, and price. These attributes contribute to a richer understanding of the data.

The next step involves the creation of dimension tables, which store the attributes of each dimension. Additionally, a fact table is designed to store the measures or numerical data points, such as sales revenue, and is connected to dimension tables through keys.
Establishing relationships and joins between the fact table and dimension tables is crucial for enabling efficient data retrieval during queries. These relationships form the backbone of the dimensional model and facilitate seamless exploration within the data cube.

Integration with Online Analytical Processing (OLAP) systems is a critical component. OLAP tools are specifically designed to work with dimensional models, providing users with a user-friendly interface for interactive analysis. This integration allows users to dynamically slice, dice, and pivot the data for different perspectives.

Flexibility in analysis paths is a key consideration. The dimensional model should allow users to drill down into specific details, roll up for summarized views, and pivot for varied analytical perspectives. This flexibility is essential for adapting to changing business needs and exploring data in diverse ways.

Regular review and optimization of the dimensional model are necessary to ensure it aligns with evolving business requirements. Adjustments may be made as new dimensions or measures become relevant, maintaining the model's effectiveness in facilitating comprehensive and efficient data analysis.

Check Your Progress:

10. What is a data cube and why is it important in data analysis?
11. List and describe the main components of a data cube.
12. What are the key characteristics of a data cube?
13. How does hierarchical organization within dimensions benefit data analysis?
14. Why is it important to regularly review and optimize the dimensional model?

4.6 Operations in OLAP

(OLAP) functions as a robust instrument for the comprehensive analysis of business data across diverse perspectives. At its core lies the pivotal construct of the OLAP cube, an intricately structured data format finely tuned for profound analysis. The cube encompasses numeric facts denoted as dimensions and is commonly referred to as 'Hyper cubes.' These structures empower users to conduct Multidimensional Analytical querying, utilizing fundamental OLAP operations like Drill-down, Roll-up, Slicing, Dicing, and Pivot. In this procedural framework, a data warehouse assumes a critical role, systematically extracting, cleansing, and transforming data from diverse sources. This meticulously refined data is subsequently loaded onto the OLAP server, culminating in a sophisticated analysis that underpins informed decision-making in business contexts.

There are five major types of OLAP operations that can be performed. These are as below:

1. Roll up
2. Drill down
3. Slice
4. Dice
5. Pivot

1. Roll up:

The roll-up operation (also known as drill-up or aggregation operation) performs aggregation on a data cube, by climbing up concept hierarchies, i.e., dimension reduction. Roll-up is like zooming-out on the data cubes. Figure shows the result of roll-up operations performed on the dimension location. When a roll-up is performed by dimensions reduction, one or more dimensions are removed from the cube.

In this example, cities New jersey and Lost Angles and rolled up into country USA. The sales figure of New Jersey and Los Angeles are 440 and 1560 respectively. They become 2000 after roll-up. In this aggregation process, data is location hierarchy moves up from city to the country. In the roll-up process at least one or more dimensions need to be removed. In this example, Cities dimension is removed.

2. Drill-Down

The drill-down operation (also called roll-down) is the reverse operation of roll-up. Drill-down is like zooming-in on the data cube. It converts the less detailed record to more detailed data. Drill-down can be performed by either stepping down a concept hierarchy for a dimension or adding additional dimensions.

Because a drill-down adds more details to the given data, it can also be performed by adding a new dimension to a cube.

Consider the diagram above. Quater Q1 is drilled down to months January, February, and March. Corresponding sales are also registers. In this example, dimension months are added.

3. Slice

A slice is a subset of the cubes corresponding to a single value for one or more members of the dimension. It performs a selection on one dimension of the given cube, thus resulting in a sub-cube.

In the example above, dimension Time is Sliced with Q1 as the filter. A new cube is created altogether.

4. Dice

The dice operation in OLAP selects a subset of data by applying conditions on multiple dimensions simultaneously. Unlike slicing, which filters data along a single dimension, dicing allows users to refine their analysis by specifying criteria across two or more dimensions. For example, in a sales data cube, dicing could involve selecting data for a specific product category, time period, and geographic region simultaneously. This operation enables users to explore detailed intersections within multidimensional data, uncovering relationships and patterns that provide deeper insights for decision-making and strategic planning in business intelligence and analytics.

5. Pivot

The pivot operation is also called a rotation. Pivot is a visualization operation which rotates the data axes in view to provide an alternative presentation of the data. It may contain swapping the rows and columns or moving one of the row-dimensions into the column dimensions.

Check Your Progress:

10. What is the primary purpose of OLAP in business data analysis?
11. Explain the concept of an OLAP cube and its significance.
12. Why is it important to have both slice and dice operations in OLAP?

1. Model is essential for organizing and analyzing large datasets in business contexts. It's widely used in data warehousing and OLAP systems, offering intuitive exploration through data cubes, dimensions, and measures.

2. Key Components: The Multidimensional Data Model consists of data cubes (visualizing data across dimensions), dimensions (categorical attributes), measures (quantitative data), fact tables (storing measures), and dimension tables (storing attributes).

3. Stages in Building the Model: The process involves assembling client data, segmenting the system, identifying dimensions and measures, preparing attributes, extracting real data, and finally, structuring the schema.

4. Features: These include efficient data analysis, intuitive representation, flexibility in analysis, fast query performance, support for complex operations, hierarchical organization, enhanced decision support, and optimization for OLAP systems.

5. Data Cubes: These are multi-dimensional arrays facilitating data modeling and analysis. They include dimensions (e.g., time, product), measures (e.g., sales), hierarchies (e.g., year, month), cells (holding data values), and cuboids (abstraction levels).

4.7 SUMMING UP

1. The Multidimensional Data Model is a powerful way for businesses to organize and analyze large amounts of data effectively. Unlike traditional databases, it lets users explore data from different angles like time, location, and product, which helps in making better decisions.

2. At the heart of Multidimensional Data Model are data cubes, which organize data into dimensions (like categories) and measures (like numbers or values). This setup allows for easy querying and interactive analysis, which is crucial in fields like Online Analytical Processing (OLAP) and data warehousing.

3. Building a Multidimensional Data Model involves collecting data, organizing it logically, and creating a structure (schema) that defines how different pieces of data relate to each other. This process ensures that data is easy to find and understand, supporting quick analysis and decision-making. Key features include fast data retrieval, flexibility in analysis, and clear visual representation, all of which help users explore and understand complex data sets.

4. OLAP operations—like roll-up (summarizing data), drill-down (getting more detailed data), slice (selecting data based on criteria), dice (analyzing data across multiple dimensions), and pivot (changing how data is viewed)—are essential tools within Multidimensional Data Model. These operations allow users to dive deep into data, see trends from different angles, and make informed choices based on what they find.

5. In summary, the Multidimensional Data Model and its OLAP tools are essential for businesses looking to make sense of large amounts of data. By learning these concepts, users can improve their analytical skills, uncover valuable insights, and contribute to better decision-making in their organizations.

4.8 Answer to Check your Progress:

1. The Multi-Dimensional Data Model is a sophisticated approach to organizing and structuring data in a database, allowing for the analysis of data from various perspectives. It uses data cubes, dimensions, and measures to enable intuitive and flexible data exploration. Unlike traditional relational databases that rely on queries for data access, the Multi-Dimensional Data Model allows users to analyze data more intuitively by posing analytical questions. It supports complex queries and provides a more user-friendly interface for exploring data in multiple dimensions.
2. Key components of the Multidimensional Data Model:

- Data Cubes: Visual representations of multidimensional relationships between measures and dimensions.
- Dimensions: Attributes that provide context to measures, such as time, location, product, or customer.
- Measures: Numerical values representing quantifiable data, like sales, revenue, or quantity.
- Hierarchies: Levels of detail within dimensions, facilitating efficient navigation and analysis.
- Fact Tables: Tables that store measures and connect to dimension tables.
- Dimension Tables: Tables that store attributes of each dimension.

3. Key features of a Multidimensional Data Model are:
    - Measures: Quantifiable data points stored in fact tables.
    - Dimensions: Attributes providing context to measures, stored in dimension tables.
    - Cubes: Structures representing multidimensional relationships, making complex datasets easier to understand.
    - Aggregation: Summarizing data across dimensions and levels of detail.
    - Hierarchies: Organizing dimensions into levels for efficient navigation and analysis.
    - OLAP Support: Optimized for Online Analytical Processing, enabling fast query response times and complex analytical operations.
4. OLAP (Online Analytical Processing) is a category of multidimensional data models designed for efficient querying and analysis of large datasets. It allows users to interactively explore and analyze data from multiple perspectives.

   OLAP is crucial for data analysis as it provides fast response times for complex queries, supports operations like drill-down, roll-up, slicing, and dicing, and enhances decision-making processes by enabling users to gain insights and perform detailed analysis on large volumes of data.

5. A data cube is a multi-dimensional array of data that allows users to model and view information from different perspectives. It is a multi-dimensional extension of two-dimensional tables in a relational database.
   Data cubes enable efficient analysis of large datasets by allowing users to quickly summarize and visualize data from multiple dimensions, facilitating better insights and decision-making.
6. The main components of a data cube are:

   - Dimensions: Represent the different facets or categories of data, such as time, geography, product, and customer.
   - Measures: Numerical data points or metrics that users want to analyze, like sales revenue or quantity sold.
   - Hierarchies: Levels of detail within a dimension, such as year, quarter, month, and day in a time dimension.
   - Cells: Represent unique combinations of dimension members and hold the measure or metric value associated with that combination.
7. The key characteristics of a data cube are:
   - Fast Query Performance: Pre-computed and stored aggregated data enables quick retrieval of summarized information.
   - Business Intelligence and Decision Support: Fundamental for OLAP systems, providing a powerful tool for interactive analysis.
   - Multidimensional Analysis: Allows users to explore data across multiple dimensions for deeper insights.
   - Aggregation and Summarization: Facilitates viewing data at different levels of granularity.

8. Hierarchical organization within dimensions benefits data analysis by allowing efficient navigation through data at various levels of granularity, such as drilling down into detailed levels or rolling up to summary levels. This structure enhances flexibility in exploration, enabling users to switch between detailed and summarized views to gain a comprehensive understanding. It also improves the ability to identify patterns and trends, as users can view data from multiple perspectives and levels of detail, facilitating more effective and insightful analysis.
9. Regularly reviewing and optimizing the dimensional model is crucial to ensure it remains aligned with evolving business requirements and provides relevant insights. This process improves performance by optimizing data retrieval and analysis efficiency. It allows for the adaptation of the model to include new dimensions or measures, maintaining its effectiveness and relevance. Additionally, regular reviews help identify and correct any inconsistencies or errors, ensuring the accuracy and quality of data analysis.
10. The primary purpose of OLAP (Online Analytical Processing) in business data analysis is to enable users to perform complex queries and multidimensional analysis efficiently. OLAP tools facilitate the exploration of large datasets from multiple perspectives, allowing users to quickly gain insights, identify trends, and make informed decisions. By providing fast, interactive access to aggregated data, OLAP enhances business intelligence and decision support processes.
11. An OLAP cube, also known as a multidimensional cube or hypercube, is a data structure that allows data to be modeled and viewed in multiple dimensions. It consists of dimensions (such as time, geography, and product) and measures (such as sales and revenue). The significance of an OLAP cube lies in its ability to provide a comprehensive and intuitive way to analyze complex data. By organizing data into

multidimensional arrays, OLAP cubes enable users to perform operations like slicing, dicing, and pivoting, facilitating in-depth analysis and insightful decision-making.

12. Having both slice and dice operations in OLAP is important because they provide different ways to interact with and analyze data within the cube. The slice operation allows users to filter the data based on a single dimension, creating a sub-cube for focused analysis. The dice operation, on the other hand, allows users to filter data across multiple dimensions, creating a more refined sub-cube for detailed analysis. Together, these operations enable users to explore data from various angles, uncovering insights that might not be apparent with a single perspective, thus enhancing the depth and flexibility of data analysis.

4.9 Possible Questions

Multiple Choice Questions

1. Which of the following is a visual representation in the Multidimensional Data Model?
   - a. Tables
   - b. Cubes
   - c. Trees
   - d. Lists

2. In the context of OLAP operations, what does "drill-down" mean?
   - a. Aggregating data
   - b. Adding more detail to data
   - c. Removing dimensions
   - d. Rotating data axes

3. What is the main advantage of using a Multidimensional Data Model in data warehousing?
   - a. Simplified data entry
   - b. Enhanced transaction processing

c. Efficient data analysis
d. Real-time data updates

4. Which component of a data cube organizes dimensions into levels of detail?
    a. Measures
    b. Hierarchies
    c. Cells
    d. Aggregation

5. Which OLAP operation involves rotating the data axes for different data presentations?
    a. Drill-down
    b. Slice
    c. Dice
    d. Pivot

True or False:

1. The Multidimensional Data Model is primarily used for managing and interpreting complex datasets in relational databases.
2. OLAP systems are designed for Online Analytical Processing, providing tools for interactive data analysis.
3. Hierarchical organization within dimensions does not provide any benefits for data analysis.
4. Data cubes allow data to be modeled and viewed in multiple dimensions, similar to two-dimensional tables in relational databases.
5. The primary purpose of OLAP in business data analysis is to manage transactional data in real-time.
6. The drill-down operation in OLAP involves aggregating data across different dimensions.
7. Dimensional modeling involves organizing data into cubes, dimensions, and measures for efficient analysis.

8. Data cubes are not used in Online Analytical Processing (OLAP) systems.
9. The Multi-Dimensional Data Model organizes data into facts, attributes, and relationships.
10. Regular review and optimization of the dimensional model are unnecessary for maintaining its effectiveness.

Fill in the blanks:

1. _________ tables store numerical values representing quantifiable data in a Multidimensional Data Model.

2. The process of summarizing data across dimensions and levels of detail is called _________.

3. _________ in OLAP is the operation that selects a sub-cube by operating a selection on two or more dimensions.

4. The visual representation of data in multiple dimensions within a Multidimensional Data Model is known as a _________.

5. The ______________operation in OLAP involves aggregating data across different dimensions.

Long Question Answer Type

1. Define what a data cube is and list its main components.
2. What is the purpose of the "roll-up" operation in OLAP?
3. Explain the term "fact table" in the context of dimensional modeling.
4. Describe the key stages involved in building a Multidimensional Data Model and explain the significance of each stage.
5. Discuss the advantages and disadvantages of the Multidimensional Data Model.
6. Explain the concept of dimension hierarchies and how they facilitate data analysis in OLAP systems.

---×---

UNIT: 5
DATA WAREHOUSING ARCHITECTURE AND DATA WAREHOUSING COMPONENTS

Unit Structure:

5.1 Introduction

In today's data-driven world, organizations rely heavily on efficient data management practices to extract insights and make informed decisions. In this environment, data warehousing is essential because it acts as a central location for organizing, storing, and analyzing large volumes of structured and unstructured data. In this chapter, we explore the foundations of data warehousing, its importance, and the key elements that make up a strong data warehousing solution.

5.2 Unit Objectives

After going through this unit you are able to:

- Gain a comprehensive understanding of the architectural framework
- Understand the roles and functionalities of each component in facilitating the flow of data within the data warehousing system.

- Understand the hardware and software components that comprise a data warehouse server infrastructure.
- Explore the concept of a virtual data warehouse
- Understand the significance of metadata in data warehousing
- Understand the functionalities and features of different access tools
- Learn about data marts as subsets of a data warehouse and explore different types of data marts
- Understand the fundamental differences between a data mart and a data warehouse
- Gain insights into the principles and practices of data warehouse management

5.3 Data Warehousing Architecture

Data warehousing architecture refers to the structure and design of a data warehouse system. It is the plan for organizing and implementing a data warehouse. It consists of many components, such as hardware, software, and processes, all of which work together to facilitate data storage, administration, and analysis. This architecture is precisely developed to accomplish many major objectives:

Data Integration: One of the primary goals of data warehousing architecture is to bring data from several sources together into a single repository. This involves collecting data from multiple operational systems, including databases, spreadsheets, and external sources, and converting it into a standardized format suitable for analysis. Integration ensures that all essential data is available in one location, allowing for analysis and decision-making.

Data Quality: Maintaining excellent data quality is critical for gaining accurate insights and making sound decisions. The data warehousing architecture includes processes and tools that ensure data correctness, completeness, and consistency. These may include data profiling, cleaning, and enrichment approaches for detecting and correcting mistakes, abnormalities, and inconsistencies in data. By enhancing data quality, the design increases the dependability of analytical results.

Easy Access to Information: One of the most important aspects of data warehousing design is to give users seamless access to information stored in the data warehouse. This involves implementing efficient information retrieval systems as well as user-friendly interfaces for accessing and evaluating data. Users should be able to get information quickly and intuitively, whether using reporting tools, analytics platforms, or direct database access.

Scalability and Performance: As data volumes increase and analytical requirements change, scalability and performance become essential factors in data warehouse design. The architecture should be built to handle rising data loads and user demands while maintaining performance. This might include using scalable hardware infrastructure, improving data storage and processing methodologies, and exploiting parallel processing capabilities to efficiently handle large-scale analytics tasks.

Security and Compliance: Data confidentiality, integrity, and availability are critical components of data warehousing architecture. Encryption, access restrictions, and audit trails are examples of security methods that assist protect sensitive information from unwanted access while also ensuring compliance with legal obligations such as GDPR, HIPAA, and industry standards. The design fosters data trust and improves overall governance by incorporating strong security measures.

NOTE: *Data warehousing architecture is a comprehensive framework that governs the design and implementation of a data warehouse system. By integrating hardware, software, and processes, it facilitates data integration, ensures data quality, and provides users with easy access to information for analytical purposes. By addressing these key objectives, data warehousing architecture enables organizations to harness the full potential of their data assets and drive informed decision-making.*

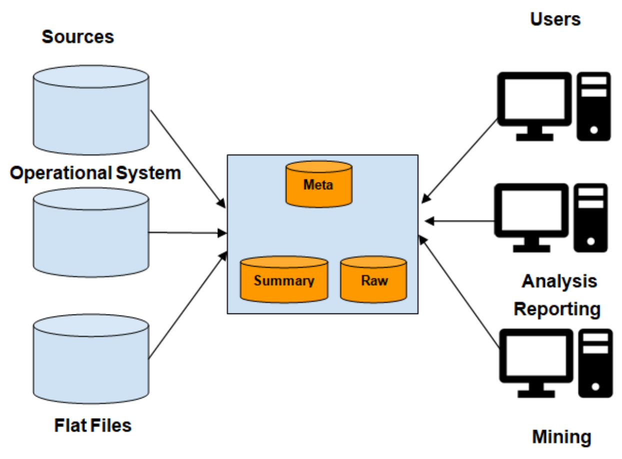


Fig: Architecture of Data Warehouse

5.4 Components of Data Warehousing Architecture

The data warehousing architecture is made up of multiple interconnected components, each of which plays an important role in the storage, administration, and analysis of data for decision-making. Let's go over these components in detail:

Source systems: Source systems are the major data repositories used to extract and load data into the data warehouse. These may include transactional databases, operational systems, CRM (Customer Relationship Management) systems, ERP (Enterprise Resource Planning) systems, spreadsheets, flat files, and external data sources. The data warehousing process begins with the extraction of data from various source systems.

ETL (Extract, Transform, Load) Processes: ETL methods collect data from source systems, turn it into an analytical format, and put it into a data warehouse. Extraction involves retrieving data from numerous source systems using methods such as bulk extraction or real-time replication. To guarantee consistency and quality, the data is cleansed, filtered, aggregated, and structured. Loading is the process of putting converted data into a data warehouse utilizing strategies like bulk loading or incremental loading.

Data Warehouse: The data warehouse stores and integrates data from several sources for analysis. It is usually tuned for read-intensive workloads and designed to handle complicated queries and data analysis operations. Dimensional modeling (e.g., star schema, snowflake schema) may be used in the data warehouse structure to improve querying and reporting efficiency.

Data marts: Data marts are subsets of a data warehouse that target certain business operations, divisions, or user groups in an organization. They include pre-aggregated and filtered data customized to the analytical requirements of certain stakeholders. Data marts can be constructed by pulling data directly from the data warehouse (dependent data marts) or by constructing them independently from the source systems.

OLAP (Online Analytical Processing) Server: OLAP servers enable users to evaluate data across several dimensions, including time, geography, and product categories. OLAP servers provide operations including slicing, chopping, drilling down, and rolling up data to get insights into corporate performance and patterns. They allow for interactive and ad hoc examination of data stored in a data warehouse or data mart.

Reporting and Analytics Tools: Reporting and analytics tools offer intuitive interfaces for searching, analyzing, and visualizing data from data warehouses or marts. These technologies allow users to build dashboards, reports, charts, and graphs to communicate information and aid decision-making processes. Reporting and analytics tools include Tableau, Power BI, Qlik View, and Micro Strategy.

Metadata Repository: A metadata repository holds information about the structure, format, lineage, and utilization of data in data warehouses or marts. It offers a consolidated metadata catalog to assist users in understanding the meaning and context of the data with which they are working. Metadata management enables data governance, data lineage tracking, impact analysis, and regulatory compliance.

Data Quality Tools: Data quality tools verify the correctness, completeness, consistency, and integrity of data held in data warehouses or marts. These technologies enhance data quality by performing activities such as profiling, cleaning, standardization, deduplication, and enrichment. Data quality tools improve analytical results by discovering and fixing mistakes, abnormalities, and inconsistencies.

Security and Access Control: Security and access control systems safeguard sensitive data kept in data warehouses or marts from unauthorized access, abuse, or breaches. These technologies consist of user authentication, role-based access control (RBAC), data encryption, row-level security, and audit trails. Security measures help to guarantee compliance with regulatory standards such as GDPR, HIPAA, SOX, and PCI-DSS.

Data Governance Framework: Data governance refers to rules, processes, and practices that ensure the quality, availability, usefulness, and security of an organization's data assets. A data governance framework establishes roles, responsibilities, and processes for data stewardship, ownership, lifecycle management, and privacy. It encourages data accountability, transparency, and compliance, hence building trust and confidence in data-driven decision-making.

***Note**: These components collectively form the architecture of a data warehouse system, providing organizations with the infrastructure and tools needed to leverage their data assets effectively for strategic insights and competitive advantage.*

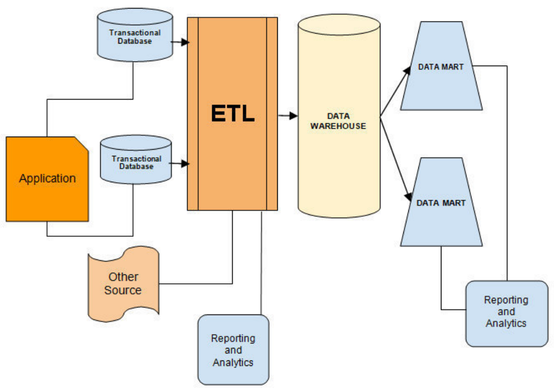


Fig: Data Warehouse Components

5.5 Data Warehouse Server

A data warehouse server is the foundation of a data warehouse system, particularly intended to manage the complicated processes of storing, processing, and providing data for analytical use. It provides a powerful and specialized computer infrastructure that is customized to the specific needs of data warehousing.

A data warehouse server may be described in detail as follows:

Specialized Computing System: A data warehouse server is designed to handle the heavy workload of a data warehouse environment. Unlike general-purpose servers, which may favor variety above performance, a data warehouse server is designed specifically for analytical processing activities. It comes with hardware components and software configurations that are specifically designed to handle enormous amounts of data and complicated analytical queries effectively.

Foundation for Storing, Processing, and Delivering Data: The fundamental function of a data warehouse server is to provide the basis for the complete data warehouse ecosystem. It offers the infrastructure required to store massive volumes of data from many sources, analyze that data to provide analytical insights, and distribute actionable information to end users. From data intake to data presentation, the data warehouse server is involved in every stage of the data lifecycle.

Storage Capabilities: A data warehouse server's capacity to store enormous information in a dependable and efficient manner is critical. It uses high-capacity storage subsystems, such as RAID arrays or storage area networks (SANs), to handle enterprises' ever-increasing data volumes. Furthermore, data storage is designed for analytical workloads, frequently employing techniques such as columnar storage or compression to improve storage economy and query performance.

Processing Power: Data warehouse servers have strong CPUs and enough memory to support analytical processing. These servers use parallel processing architectures and specialized query execution engines to efficiently run complicated analytical queries on massive datasets. Data warehouse servers can provide quick query response times while also supporting concurrent user access by dividing query workloads across numerous CPU cores.

Analytical Capabilities: Data warehouse servers generally provide analytical functionality with storage and processing capabilities. These might include sophisticated SQL capabilities, in-database analytics, interaction with machine learning libraries, and data mining methods. Organizations may expedite their analytical workflow and gain important insights from their data by integrating analytical capabilities right into the data warehouse server.

Scalability and Flexibility: Data warehouse servers can extend vertically and horizontally to meet changing data volumes and user needs. Vertical scaling entails updating hardware components, such as adding additional CPU cores or memory modules, to boost the server's processing capacity. Horizontal scalability is adding more servers to distribute workload over a cluster of machines, resulting in linear scalability and fault tolerance.

Reliability and availability: are critical in a data warehouse setting, as downtime may have serious economic consequences. Data warehouse servers are designed to be highly available, with redundancy, failover procedures, and data replication to reduce the risk of service outages. Furthermore, proactive monitoring and management technologies are frequently used to detect and fix possible problems before they affect system performance.

Note: *A data warehouse server is a customized computer system that serves as the foundation for a data warehouse environment. It offers the necessary infrastructure for storing, processing, and disseminating data for analytical purposes, allowing enterprises to derive important insights and make educated decisions.*

5.6 Virtual Data Warehouse

A virtual data warehouse (VDW) is a conceptual framework or architecture that allows companies to access and analyze data from several sources without physically consolidating it into one repository. Unlike traditional data warehouses, which need costly data integration and storage processes, a virtual data warehouse use virtualization technique to deliver a unified view of data across several systems.

In a virtual data warehouse, data is still scattered among many source systems, including databases, cloud platforms, and external data sources. Instead of physically transferring or copying data into a centralized repository, the virtual data warehouse develops a logical layer that abstracts the underlying data sources and exposes them to users and applications as a single, unified dataset.

Virtualization technologies such as data federation, data abstraction, and data virtualization platforms are critical for allowing virtual data warehouses. These technologies enable users to access and evaluate data from numerous sources in real time, eliminating the need for data migration or duplication.

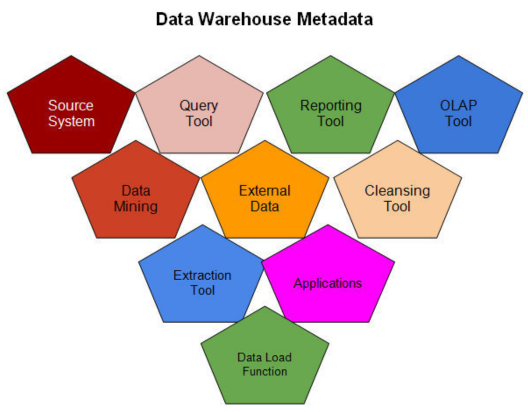
Organizations that employ a virtual data warehouse method can overcome many of the issues associated with traditional data warehousing, such as data integration complexity, storage costs, and data latency. Virtual data warehouses provide flexibility, agility, and scalability, allowing enterprises to respond swiftly to changing business requirements and easily integrate new data sources.

Furthermore, virtual data warehouses help to democratize data by allowing business users, data analysts, and data scientists to access a variety of data sources on their own. Users may explore, analyze, and display data from a variety of sources using familiar tools and interfaces, without needing a specific understanding of the underlying data structures or technology.

Overall, a virtual data warehouse is a contemporary approach to data management and analytics, allowing companies to better exploit their different data assets and generate actionable insights to support informed decision-making.

5.7 Metadata

Metadata, in the context of a data warehouse, is an important layer of descriptive information about the underlying data assets housed within the system. It includes facts about the data's source, structure, relationships, and business context, giving users significant insights into its sources, organization, and significance. Metadata helps data discovery, analysis, and governance by providing a full perspective of the data landscape as well as guaranteeing compliance with regulatory requirements and corporate norms. Furthermore, metadata is critical to data integration and interoperability, allowing for the easy interchange of data between diverse systems and platforms inside the data warehouse. Overall, metadata serves as a foundation for efficiently analyzing, maintaining, and using data assets for analytical purposes inside the data warehouse ecosystem.



Check Your Progress:

1. What is metadata, and why is it important in a data warehouse environment?
2. What types of information does metadata typically encompass in a data warehouse?
3. How does metadata facilitate data discovery and interpretation within a data warehouse?
4. In what ways does metadata enhance the reliability and trustworthiness of data stored in a data warehouse?
5. Can you explain the role of metadata in data governance within a data warehouse ecosystem?

5.8 Access Tools

As we all know, a data warehouse is a data management system that stores, reports, and analyzes data. It is the core component of business intelligence, commonly referred to as an enterprise data warehouse. Data warehouses are a central repository for data from one or more diverse sources. Data warehouses are analytical tools designed to aid decision-making for reporting users from several departments. A data warehouse creates a single, uniform system of truth for a whole company by storing historical data about businesses and organizations so that it can be studied and insights extracted from it.

Data warehouses and technologies are transitioning from physical data centers to cloud-based data warehouses. Many major firms continue to employ traditional data warehousing methods, but the future of the data warehouse is unquestionably cloud-based. Cloud-based data warehousing technologies are quick, efficient, scalable, and pay-per-use.

There are several cloud-based data-warehousing options available. As a result, it is difficult to identify the best data Warehouse tools for our project's needs. Here are the top 5 data warehousing tools:

1. **Microsoft Azure:** Microsoft released Azure, a cloud computing platform, in 2010. Microsoft Azure is a cloud computing service that allows you to create, test, deploy, and manage applications and services using Microsoft-managed data centers. Azure is a public cloud computing platform that provides infrastructure as a Service (IaaS), PaaS, and SaaS. The Azure cloud platform offers over 200 products and cloud services, including Data Analytics, Virtual Computing, Storage, Virtual Network, Internet Traffic Manager, Web Sites, Media Services, Mobile Services, Integration, and more. Azure enables seamless mobility and a truly compatible platform across on-premises and public clouds. Azure offers a variety of cross-connections, including virtual private networks (VPNs), caches, content delivery networks (CDNs), and Express Route connections, to increase usability and speed. Microsoft Azure provides a secure foundation for physical infrastructure and operational security. Azure App provides a fully managed web hosting

solution for developing web apps, services, and Restful APIs. It provides a range of plans to satisfy the needs of every application, from modest to global-scale web applications. One of Microsoft Azure's most popular uses is the cloud-based deployment of virtual machines and containers.

2. **Google Big Query:** Google Big Query is a cloud-based data warehouse powered by the Google Cloud Platform. It is built to manage large-scale data analytics tasks, allowing businesses to store, query, and analyze big datasets quickly and affordably. Big Query uses a server less paradigm, which eliminates the requirement for users to deploy and manage infrastructure. It employs a distributed design that dynamically scales resources in response to workload needs, resulting in consistent performance independent of query complexity or dataset size. Data is stored in columnar format, which improves compression and query performance. Big Query works smoothly with other GCP services like cloud storage, dataflow, and the AI platform, allowing customers to create end-to-end data pipelines and conduct complex analytical operations. It has built-in support for sophisticated analytics like machine learning and geospatial analysis.

3. **Snowflake:** Snowflake is a cloud-native data warehousing software notable for its distinct design that separates storage and processing. This separation allows for automated scalability and concurrency, making it extremely adaptable and performant. Snowflake handles both structured and semi-structured data, with capabilities including data sharing, data lake connection, and support for several data types. It has a variety of built-in features for data processing, analytics, and machine learning. Snowflake's pricing approach is based on real utilization, with choices for on-demand and pre-purchased capacity, allowing customers to efficiently control expenditures. It has strong security features including encryption, access restrictions, and compliance certifications to ensure data privacy and regulatory compliance. Snowflake's cloud-agnostic strategy enables customers to create data warehouses across various

cloud providers, increasing flexibility while avoiding vendor lock-in.

4. **Amazon Redshift:** Amazon Redshift is a fully managed data warehouse service offered by Amazon Web Services (AWS). It is built to handle large-scale analytics workloads, allowing businesses to store and analyze massive volumes of data fast and affordably. Redshift uses a massively parallel processing (MPP) architecture to distribute data and query execution over numerous nodes, resulting in great performance and scalability. It supports both structured and semi-structured data types, enabling users to evaluate a wide range of information. Redshift works smoothly with other AWS services including S3, Dynamo DB, and IAM, allowing users to create extensive data pipelines and execute complex analytics. It includes features like automated backups, snapshots, and encryption to assure data security and compliance with regulations.

5. **IBM Db2 Warehouse on Cloud:** IBM Db2 Warehouse on Cloud is a fully managed cloud-based data warehousing solution available through IBM Cloud. It is intended to serve analytics workloads by offering a scalable and high-performance environment for storing and analyzing massive amounts of data. Db2 Warehouse on Cloud is based on IBM's Db2 database technology and is designed specifically for analytics and data warehousing applications. It supports structured and semi-structured data types, allowing users to consume and analyze a wide range of information. Db2 Warehouse on Cloud is a completely managed service that eliminates the need for customers to handle infrastructure, such as provisioning, scaling, and maintenance chores. This enables businesses to concentrate on data analysis and insight development without worrying about the underlying infrastructure.

Check Your Progress:

6. What are the benefits of using cloud-based data warehousing tools compared to on-premises solutions?
7. Can you explain how cloud-based data warehousing tools handle scalability?
8. What factors should organizations consider when choosing a cloud-based data warehousing tool?
9. How do cloud-based data warehousing tools ensure data security and compliance?
10. What are some common use cases for cloud-based data warehousing tools?

5.9 Data Marts

Data marts are specialized subsets of a data warehouse created to meet the needs of certain business units, departments, or user groups inside an organization. Unlike data warehouses, which hold integrated and consolidated data from diverse sources throughout the organization, data marts include data that is specialized to fulfill the analytical and reporting needs of a specific business sector.

Data marts are commonly used to give faster and more efficient access to important data for decision-making and analysis in specialized fields. They provide a more concentrated and simplified approach to data storage and retrieval, allowing users to get the information they want without having to trawl through large volumes of irrelevant material.

5.9.1 Types of Data Marts

Dependent Data Marts: These data marts are created directly from the core data warehouse. They are monitored and managed centrally, and the data in these marts comes from the data warehouse. Dependent data marts are commonly employed when individual business units or departments want access to a portion of the data warehouse's data that is suited to their needs.

Independent data marts: are created separately from the data warehouse and may incorporate data from many operating systems or external sources. These data marts are often designed to meet the analytical requirements of a single business unit or user group, rather than depending on a centralized data warehouse. Independent data marts provide more flexibility and autonomy, but they may produce duplicate or incorrect data if not properly coordinated with the data warehouse.

5.9.2 Difference between Data Mart and Data Warehouse

Data Mart	Data Warehouse
Focus: A single subject or functional organization area	Focus: Enterprise-wide repository of disparate data sources
Data Sources: Relatively few sources linked to one line of business	Data Sources: Many external and internal sources from different areas of an organization
Size: Less than 100 GB	Size: 100 GB minimum but often in the range of terabytes for large organizations
Normalization: No preference between a normalized and denormalized structure	Normalization: Modern warehouses are mostly denormalized for quicker data querying and read performance
Decision Types: Tactical decisions pertaining to particular business lines and ways of doing things	Decision Types: Strategic decisions that affect the entire enterprise

Check Your Progress:

1. What is a data mart, and how does it differ from a data warehouse?
2. Why would an organization choose to implement data marts in addition to a data warehouse?
3. Can you explain the concept of dependent and independent data marts?
4. How do data marts contribute to improved decision-making within organizations?
5. What are the key considerations when designing and implementing data marts?

5.10 Data Warehouse Management

Data warehouse management is the design, development, implementation, maintenance, and optimization of a data warehouse environment to ensure that it meets an organization's data management and analytics requirements. Here's a summary of what data warehouse management includes:

1. **Planning and Design:** This includes determining the organization's business needs, identifying data sources, creating data models, and developing the data warehouse's architecture. It entails deciding what data will be saved, how it will be arranged, and how it will be retrieved.

2. **Development and Implementation:** Once the planning and design stages are completed, the data warehouse must be created and executed. This involves data extraction, transformation, and loading (ETL), data model creation, database and table design, and data access and analytics interface development.

3. **Data Integration:** Data warehouse management is bringing together data from several sources, which may include operational databases, external data sources, spreadsheets, and other data repositories. This procedure necessitates maintaining data quality, consistency, and integrity across all connected data sources.

4. **Data Security and Governance:** Managing data warehouse security include safeguarding sensitive data, restricting access to the data warehouse, and ensuring compliance with regulatory standards such as GDPR, HIPAA, and SOX. Data governance mechanisms must also be implemented to assure data quality, consistency, and compliance with business policies.
5. **Performance Tuning and Optimization:** Data warehouse management is monitoring the performance of the data warehouse environment, finding bottlenecks or performance concerns, and optimizing queries, indexes, and data structures to optimize performance and scalability.

6. **Backup and Disaster Recovery:** Data warehouse management includes developing backup and disaster recovery plans to assure data availability and integrity in the case of hardware failures, data corruption, or other calamities. This includes frequent backups, data replication, and failover procedures.

7. **Metadata Management:** Managing metadata is an important part of data warehouse management. This entails documenting data definitions, data lineage, data transformations, and other metadata properties in order to offer context and understanding of the data in the warehouse.

Check Your Progress:

6. What is data warehouse management, and why is it important for organizations?
7. Can you explain the process of planning and designing data warehouse architecture?
8. What are some common challenges faced during the implementation phase of data warehouse management?
9. How do organizations ensure data security and governance in a data warehouse environment?
10. What strategies can be implemented to optimize query performance and improve efficiency in a data warehouse?

5.11 Summing Up

- Metadata in a data warehouse contains descriptive information about the underlying data assets, such as their source, structure, and connections.

- It gives useful information and insights regarding the data's origins, allowing users to assess its dependability, trustworthiness, and usefulness.

- Metadata aids data discovery by providing information on the data's organization, schema, tables, and fields, allowing users to browse and interpret the data structure more efficiently.

- ❖ In addition to structural features, metadata contains semantic information about the data, such as its meaning, definitions, and business context, which improves its interpretability and relevance.

- ❖ Metadata acts as a store for policies, regulations, and guidelines controlling data maintenance and consumption, guaranteeing compliance with regulatory requirements and corporate norms.

- ❖ It contributes significantly to data governance by documenting data governance frameworks, access restrictions, and compliance requirements inside the data warehouse environment.

- ❖ Metadata promotes data integration and interoperability by providing a standardized framework for characterizing data assets, allowing for smooth exchange and interoperability across multiple systems and platforms.

❖ Overall, metadata serves as a foundation for efficiently analyzing, maintaining, and using data assets for analytical purposes inside the data warehouse ecosystem.

❖ **Scalability:** Cloud-based data warehousing technologies enable enterprises to flexibly scale their data architecture up or down based on demand, eliminating the need for an upfront hardware investment.

- ❖ **Flexibility:** These technologies offer storage and computing resource flexibility, allowing customers to adapt to changing business demands and workloads.

- ❖ **Cost-effectiveness:** Cloud-based data warehousing technologies often work on a pay-as-you-go basis, allowing enterprises to pay only for the resources they utilize, resulting in cost savings over traditional on-premises systems.

- **Performance:** Cloud-based data warehousing applications frequently employ distributed computing and parallel processing to provide excellent performance, allowing users to examine enormous amounts of data rapidly and effectively.

- **Integration:** These technologies work seamlessly with other cloud services and data sources, allowing enterprises to create end-to-end data pipelines and perform complex analytics.
- **Security:** Cloud-based data warehousing technologies offer strong security features like encryption, access restrictions, and compliance certifications to safeguard data at rest and in transit.

- **Managed Services:** Many cloud-based data warehousing products are completely managed, which relieves enterprises of the burden of infrastructure management and allows them to focus on data analysis and insight development.

- **Collaboration:** Cloud-based data warehousing technologies frequently include collaboration features, which allow several people to view and analyze data at the same time, regardless of location.

- Data marts are subsets of data warehouses that include data particular to a business unit or department.

- **Tailored to Certain Needs:** They are intended to fulfill the analytical and reporting requirements of certain business sectors, giving users quick access to pertinent data.

- Data marts provide for quicker data access and analysis since they store only the data required for certain tasks, as opposed to searching the full data warehouse.

- Data marts are classified into two types: dependent and independent. Dependent data marts are generated directly from the data warehouse, whereas independent data marts are created independently.

- **Control and management:** Dependent data marts are centrally controlled and managed, with data coming from the data warehouse. Independent data marts are created independently and may include data from several sources.

- ❖ **Flexibility vs. Consistency:** While independent data marts provide greater freedom, they may produce duplicate or inconsistent data if not properly coordinated with the data warehouse.

- ❖ **Scalability and Agility:** Data marts can be produced and adjusted fast, allowing businesses to respond to changing business objectives and analytical demands.

- ❖ **Improved Decision-Making:** By delivering concentrated and relevant data, data marts facilitate improved decision-making and analysis within certain business areas.

5.12 Answer to Check your Progress

1. The source, structure, and meaning of data assets are all described in detail in metadata found in a data warehouse. For the purpose of comprehending the origins of data, promoting data governance, easing data discovery, and improving interpretability, it is essential. All things considered, metadata makes it possible to manage data in a warehouse setting in an effective and efficient manner.

2. A data warehouse's metadata usually contains a variety of information kinds, such as:
   Structural Information: Specifics regarding the arrangement of the data, including tables, columns, schema, and connections between various data items.
   Source Information: Details regarding the data's system of origin, data extraction techniques, and frequency of data refreshes.
   Data Definitions: Meanings, types, and any applied transformations of the data pieces are defined and described.
   Business Context: Details regarding the business context of the data, such as its applicability, significance, and role in the decision-making process.
   Data quality metrics: These are measurements of the

correctness, timeliness, completeness, and consistency of data.
Information on security guidelines, permissions controlling who can view and alter data, and access controls.

3. Through data organization, definitions and descriptions, searchability, relationship mapping, usage insights, data lineage tracking, and business context, metadata makes data discovery and understanding easier within a data warehouse.

4. By providing users with the capacity to evaluate credibility and make well-informed decisions, metadata improves the dependability and trustworthiness of data in a data warehouse by recording its sources, quality metrics, lineage, usage statistics, compliance, and business context.

5. By establishing governance frameworks, imposing access rules, collecting compliance requirements, controlling data quality, tracking data lineage, keeping audit trails, and allowing data stewardship and ownership, metadata in a data warehouse ecosystem helps data governance.

6. Cloud-based data warehousing solutions are a preferred option for organizations looking to modernize their data infrastructure because they offer features like scalability, cost efficiency, ease of deployment, accessibility, built-in security, advanced analytics capabilities, automatic updates and maintenance, and global reach compared to on-premises solutions.

7. Scalability is handled by cloud-based data warehousing tools through elastic resources, auto-scaling, compute and storage separation, pay-as-you-go pricing and serverless computing options. These features allow organizations to efficiently and economically adjust resources based on workload demands.

8. Organizations should think about scalability, performance, cost, integration, security, user-friendliness, data processing capabilities, data governance features, vendor reputation, and future flexibility when selecting a cloud-based data warehousing platform. These elements guarantee that the

chosen tool satisfies the needs of the company for effective analytics, data management, and decision-making.

9. Through encryption, access controls, authentication, audit trails, data masking, compliance certifications, data governance frameworks, frequent security upgrades, and patch management, cloud-based data warehousing solutions guarantee data security and compliance. These safeguards decrease security risks, preserve regulatory compliance, and safeguard sensitive data.

10. Business intelligence, data warehousing, operational reporting, data exploration, predictive analytics, customer analytics, financial planning, supply chain optimization, healthcare analytics, compliance, and regulatory reporting are just a few of the uses for cloud-based data warehousing tools that are used across industries. Organizations may centralize data, get insights, streamline processes, and promote well-informed decision-making with the help of these tools.

11. A data mart is an organized subset of a data warehouse that concentrates on particular departments or subject areas inside a company. A data mart provides customized datasets and analytics capabilities, catering to the demands of certain user groups or business operations, whereas a data warehouse unifies data from throughout the whole organization.

12. Alongside a data warehouse, data marts are used by enterprises to meet departmental needs, maximize performance, empower users, improve data security, personalize reporting and analysis, grow gradually, and cut costs.

13. A central data warehouse shares its data and integration procedures with dependent data marts. Independent data marts are built independently, straight from source systems or feeds, and customized to meet departmental requirements.

14. Through the provision of targeted data, timely insights, self-service analytics, tailored reporting, increased accuracy,

greater collaboration, agile response to changes, and performance optimization within particular departments or business functions, data marts facilitate better decision-making.

15. Understanding business requirements, locating data sources, creating data models, putting data integration procedures into place, establishing data governance, maximizing performance, guaranteeing scalability and flexibility, controlling user access and security, upholding metadata, and offering user support and training are all important factors to take into account when creating and executing data marts. These elements aid in the creation of solid data mart solutions that enable efficient decision-making procedures and are in line with corporate requirements.

16. The administration of data warehouses entails supervising the development, execution, upkeep, and enhancement of a centralized data repository to guarantee data security, performance, quality, and integration. It is essential for businesses because it facilitates advanced analytics and reporting, improves data integrity, scalability, and compliance, and helps well-informed decision-making.

17. Defining business requirements, locating data sources, modeling data, creating ETL processes, selecting hardware and infrastructure, deciding on an architecture type, putting data governance and security measures in place, managing metadata, maximizing performance, and offering user support and training are all part of planning and designing a data warehouse architecture. These procedures guarantee that the architecture of the data warehouse is in line with business requirements, facilitates well-informed decision-making, preserves data quality, and permits sophisticated analytics.

18. Organizations frequently encounter difficulties with data integration complexity, performance optimization, data quality assurance, scalability and flexibility, data governance and security, change management, resource limitations, user adoption and training, technical complexity, and calculating

success and return on investment during the implementation phase of data warehouse management. Proactively addressing these issues is crucial to the implementation's success and to optimizing the data warehouse's value.

19. Organizations use access controls, authentication, encryption, data masking, auditing, monitoring, data loss prevention (DLP), compliance management, employee training, and awareness initiatives to guarantee data security and governance in a data warehouse environment. These safeguards preserve regulatory compliance, guard confidential information, stop unwanted access or disclosure, and promote an organization-wide data security culture.

20. Organizations can utilize techniques like data modeling, indexing, partitioning, compression, aggregation, query optimization, concurrency control, hardware optimization, data caching, and query rewriting to maximize query performance and efficiency in a data warehouse. These actions boost system performance overall, decrease resource use, and speed up data retrieval.

5.13 Possible Questions

5.13.1 Multiple Choice Questions:

1. What is the primary goal of data integration in data warehousing architecture?
   A) Ensuring data security
   B) Bringing data from multiple sources into a single repository
   C) Facilitating data storage
   D) Improving data quality

2. Which component of data warehousing architecture focuses on processes and tools to ensure data correctness, completeness, and consistency?
   A) Data Integration
   B) Easy Access to Information
   C) Data Quality
   D) Scalability and Performance

3. Which cloud computing platform introduced Microsoft Azure?
   A) Microsoft
   B) Google Cloud Platform (GCP)
   C) Amazon Web Services (AWS)
   D) IBM Cloud
4. How does a virtual data warehouse differ from a traditional data warehouse?
   A) It requires costly data integration and storage processes
   B) It physically consolidates data into one repository
   C) It relies on manual data migration and duplication processes
   D) It uses virtualization techniques to deliver a unified view of data across multiple systems
5. What is the primary function of a data warehouse server?
   A) Processing transactions
   B) Providing cloud storage
   C) Storing, processing, and delivering data for analytical use
   D) Hosting web applications

5.13.2 Fill in the Following Blanks:

1. Data warehousing architecture aims to bring data from several sources together into a single repository through the process of ____________.

2. The architecture includes processes and tools to ensure data correctness, completeness, and consistency, collectively known as ____________.

3. Providing users with seamless access to information stored in the data warehouse is a key aspect of enhancing ____________.

4. As data volumes increase and analytical requirements change, ensuring ________ becomes essential in data warehouse design.

5. Data warehouse servers leverage strong CPUs, enough memory, and parallel processing architectures to efficiently run complicated analytical queries on massive datasets, ensuring quick query response times and supporting concurrent user access by dividing query workloads across numerous CPU ___________.

5.13.3 State Whether True or False

1. Data warehousing architecture primarily focuses on the physical storage of data.

2. Data integration in data warehousing architecture involves collecting data from multiple sources and converting it into a standardized format suitable for analysis.

3. Scalability and flexibility are not important features of data warehouse servers, as they are designed for fixed workloads.

4. Virtual data warehouses provide limited flexibility, agility, and scalability compared to traditional data warehouses.

5. IBM Db2 Warehouse on Cloud requires customers to handle infrastructure tasks such as provisioning and maintenance.

5.13.4 Short Answer Questions

1. What is the primary function of a data warehouse?
2. How does a virtual data warehouse differ from a traditional data warehouse?
3. What are some benefits of transitioning to cloud-based data warehousing technologies?
4. Briefly explain the concept of serverless paradigm in the context of Google BigQuery.
5. Describe one notable feature of Snowflake's architecture that sets it apart from traditional data warehousing solutions.

5.13.5 Long-Answer Questions

1. Discuss the role of a data warehouse in modern data management systems. Explain its significance in facilitating data storage, reporting, and analysis for businesses.

2. Compare and contrast traditional data warehousing methods with cloud-based solutions, examining scalability, cost, and management ease.

3. Explain the concept of virtual data warehouses (VDWs), including their functioning.

4. Analyze the key features and capabilities of leading cloud-based data warehousing tools. Compare their architectures, pricing models, and integration capabilities, and assess their suitability for different business needs.

5. Evaluate the future trends and advancements in data warehousing technologies. Consider how these innovations may address current challenges and shape the future of data management and analytics.

5.13ANSWERS TO CHECK YOUR PROGRESS

5.12. 1	1. (B) 2.(C) 3.(A) 4.(D) 5. (C)
15.12.2	**1.** Data Integration **2**.Data Quality **3**.Easy Access to Information **4.** Scalability and Performance **5.** Cores
15.12.3	1.False 2. True 3. False4. False5. False

---×---

UNIT: 6
OLAP ENGINE AND DATA WAREHOUSE BACKEND PROCESS

Unit Structure:

6.1 Introduction

In the field of business intelligence and data management, OLAP (Online Analytical Processing) engines and data warehouse backend procedures play critical roles. OLAP engines support multidimensional analysis, allowing users to interact with and understand data from multiple perspectives. They handle sophisticated inquiries and deliver quick insights, which is critical for strategic decision-making. Data warehouses, on the other hand, function as centralized stores for large amounts of structured data, assuring its availability and integrity. Data warehouses' backend procedures include data extraction, transformation, loading, and management, which all work together to ensure that data is accurate, integrated, and easy to analyze.

This chapter goes into the mechanics of ROLAP and MOLAP engines, explains the comprehensive backend processes of data warehouses, and analyzes their practical applications in improving company operations and decision-making.

6.2 Unit Objectives

After going through this unit, you are able to:

- Define Online Analytical Processing (OLAP) and explain its role in data analysis.
- Differentiate between Relational OLAP (ROLAP) and Multidimensional OLAP (MOLAP), including their architectures, benefits, and drawbacks.
- Describe the data storage and query execution processes in ROLAP.
- Explain how MOLAP stores and processes data in multidimensional cubes.
- Analyze the use cases where ROLAP or MOLAP is more appropriate based on data characteristics and analysis requirements.
- The steps involved in the data warehouse backend process, such as data extraction, transformation, loading (ETL), and data maintenance.

- Identify strategies for efficient data storage, such as partitioning and indexing.
- State the practical applications of OLAP and data warehousing in business intelligence, such as reporting, dashboards, trend analysis, and forecasting.
- Assess the scalability of various OLAP engines and data warehouse systems in order to meet increasing data volumes and user needs.

6.3 Online Analytical Processing (OLAP)

Online Analytical Processing (OLAP) is a powerful data analysis system that allows users to engage with and study multidimensional data from numerous perspectives. It is intended to handle sophisticated queries and deliver quick insights, making it a critical component of business intelligence (BI) systems. OLAP enables users to execute complex computations, trend analysis, and data modeling, supporting informed decision-making within an organization.

6.3.1 Characteristics of OLAP

Online Analytical Processing (OLAP) systems are intended to support complicated analysis and query processing on massive datasets, making them an essential tool in corporate intelligence. The main properties of OLAP systems are:

- **Multidimensional data model**

  Data Cubes: OLAP systems aggregate data into multidimensional structures known as data cubes, with each dimension representing a different property (for example, time, region, product).

Dimensions and Hierarchies: Dimensions frequently include hierarchies that allow for various levels of granularity. Years, quarters, months, and days are some examples of temporal dimensions.

- **Complex query processing**

  Aggregation: OLAP can perform complex aggregations and summarizations such sums, averages, counts, and other statistical operations. OLAP provides complicated aggregations and summarizations, including sums, averages, counts, and other statistical operations.

- **Derived Measures:** Calculations and derived measures can be generated dynamically, allowing for complex data analysis.

- **Interactive Data Exploration:**

  Drill-Down and Roll-Up: Users may explore the data at various degrees of detail. Drill-down gives more specific information, whereas roll-up aggregates data at higher levels.

  Slice and Dice: Users may filter and view data from various angles by slicing (choosing a subset of data) and dicing (rearranging dimensions).

- **High Performance:**

  Pre-Aggregation: OLAP systems frequently pre-aggregate data, resulting in much faster query response times.

  Indexing and Caching: Advanced indexing and caching techniques are used to improve speed and provide rapid access to frequently requested material.

6.3.2 Types of OLAP

Online Analytical Processing (OLAP) systems are classified according to their underlying architecture and how they store and manage data. The three main forms of OLAP are Relational OLAP (ROLAP), Multidimensional OLAP (MOLAP), and Hybrid OLAP (HOLAP). Each variety has various characteristics, benefits, and applications.

1. **Relational OLAP (ROLAP):**

   Relational OLAP (ROLAP) stores and manages data in relational databases, with query processing performed using a relational database management system (RDBMS).

   Key Characteristics:

   - **Data Storage:** Data is organized in relational tables.
   - **Query execution:** the process of retrieving and analyzing data using SQL queries.
   - **Dynamic Data:** Suitable for contexts where data is regularly updated.

   Advantages:

   - **Scalability:** Relational databases' scalability allows them to manage massive amounts of data.
   - **Flexibility:** Compatible with current relational databases, avoiding the requirement for specialist OLAP servers.
   - **Integration:** Works well with other database apps and utilities.

   Disadvantages:

   - **Performance:** May be slower for sophisticated queries than MOLAP since it depends on SQL for multidimensional analysis.
   - **Complexity:** Creating efficient schemas and queries might be challenging.

   Use Cases of ROLAP

   Relational OLAP (ROLAP) solutions perform particularly well in contexts with massive datasets and dynamic data.

One common use for ROLAP is in large-scale data warehouses, where enterprises must store and analyze vast amounts of data. For example, e-commerce organizations frequently use ROLAP to analyze years of consumer transactions, product sales, and website interactions, taking advantage of relational databases' scalability.

ROLAP is also appropriate for instances in which data is often changed. Financial firms, for example, must store real-time transaction data, stock market data, and client account changes, necessitating a system that can handle frequent data refresh cycles effectively.

ROLAP also excels at complex queries and ad hoc analysis. Healthcare companies, for example, utilize ROLAP to evaluate patient information, treatment outcomes, and operational indicators, relying on its capacity to perform complicated SQL queries and provide thorough reports.

ROLAP is useful in scenarios needing data integration from different sources because of its ability to smoothly interface with various relational databases and external data sources. Retail chains may utilize ROLAP to aggregate sales data, inventory records, and customer input from many regional locations, allowing for more thorough data analysis for inventory management and customer service enhancement.

ROLAP is also extremely useful for compliance and audit reporting, especially in businesses such as pharmaceuticals that require extensive tracking and reporting for regulatory reasons. Similarly, in customer relationship management

(CRM), firms utilize ROLAP to evaluate client data, purchase history, and interaction patterns, hence improving consumer segmentation and tailored marketing.

ROLAP is also used extensively in supply chain and logistics management. Manufacturing companies, for example, use production data, supplier delivery dates, and inventory turnover rates to improve supply chain operations and cut costs.

Financial and risk analysis are other significant use cases for ROLAP. ROLAP is used by investment banks to undertake complicated financial modeling, budgeting, and forecasting, as well as to analyze large datasets of financial transactions and market data to guide investment strategies and risk-management policies.

Finally, ROLAP is utilized to track operational performance across several business functions. The hotel business, for example, monitors key performance measures like as occupancy rates, guest satisfaction scores, and operating expenses to improve service quality and profitability.

2. **Multidimensional OLAP (MOLAP)**

   Multidimensional OLAP (MOLAP) stores data in specialized multidimensional databases (also known as data cubes) in a manner that is geared for speedy query processing.

Key Characteristics:

- **Data Storage:** Data is stored as multidimensional cubes.
- **Query Execution:** Optimized for fast querying and aggregation.
- **Performance:** Performance is often quicker than ROLAP because to pre-aggregation and specific storage architectures.

Advantages:

- **Performance:** Pre-aggregated data and efficient storage contribute to high query performance.
- **Ease of Use:** Simplifies difficult query processes while providing intuitive multidimensional displays.
- **Speed:** Interactive data analysis requires rapid reaction times.

Disadvantages:

- **Scalability:** Due to the requirement for pre-aggregation, the system may struggle with very big datasets.

- **Storage:** Pre-aggregated data may demand a substantial amount of storage space.

- **Data Refresh:** Updating data might be more difficult and time-consuming than ROLAP.

Use cases:

Multidimensional OLAP (MOLAP) systems perform well in settings demanding quick query speed and interactive data processing. MOLAP is utilized in financial institutions for real-time data analysis, as well as retail chains for trend analysis. It allows extensive financial modeling and forecasting in investment banks, as well as consumer segmentation and tailored marketing in CRM systems. MOLAP is also useful for optimizing supply chains in industrial companies and tracking operational performance in a variety of industries, including hospitality.

3. **Hybrid OLAP(HOLAP):**

Hybrid OLAP (HOLAP) combines parts of ROLAP and MOLAP to take use of both methodologies while minimizing their drawbacks.

Key Characteristics:

- **Data storage:** Data storage options include relational tables and multidimensional cubes.
- **Query Execution:** Can switch between ROLAP and MOLAP query processing depending on the type of the query.
- **Flexibility and Performance:** Achieves a balance between ROLAP scalability and MOLAP performance.

Advantages:

- **Flexibility:** Can select the optimum storage and query execution mechanism based on the query.
- **Performance:** Increases performance by combining pre-aggregated data with the capacity to query comprehensive relational data.
- **Scalability:** Higher scalability than pure MOLAP systems.

Disadvantages:

- **Complexity:** Implementation and administration might be more difficult owing to the multiple storage and processing techniques.
- **Cost:** Maintaining both relational and multidimensional databases may result in greater expenditures.

Use Cases:

- Organizations that require MOLAP speed for regularly requested data while simultaneously managing big and dynamic datasets, such as ROLAP.
- Environments in which various sorts of queries (both basic and complicated) are executed on a regular basis.

Notes: *Understanding the various types of OLAP systems allows firms to select the design that best meets their individual data analysis and operational requirements. Each kind has various advantages and disadvantages; therefore, it is critical to evaluate the nature of the data, the complexity of the queries, and the performance requirements before choosing an OLAP solution.*

6.4 Difference between Relational OLAP (ROLAP) and Multidimensional OLAP (MOLAP)

Aspect	**ROLAP**	**MOLAP**
Data Storage	Relational databases (tables)	Multidimensional cubes
Query Execution	Relies on SQL queries	Optimized for multidimensional queries
Architecture	Utilizes existing relational database	Specialized multidimensional data storage

Benefits	Flexibility, Scalability, Dynamic Data	Performance, Simplicity, Aggregation
Drawbacks	Slower for complex queries, Storage overhead, Complexity	Scalability, Storage requirements Data refresh

This table summarizes the differences between ROLAP and MOLAP in several areas, including data storage, query execution, architecture, advantages, and downsides.

6.5 Data Storage in ROLAP

Relational OLAP (ROLAP) systems store data in relational databases, taking use of the tables and connections found in relational database management systems (RDBMS). The data is structured into tables that represent dimensions and facts, much like a standard relational database.

Dimension tables: Dimension tables include descriptive qualities that offer context for the data being studied. Each dimension table usually represents one component of the data, such as time, region, or product categories. These tables are frequently standardized to eliminate redundancy and enhance data integrity.

Fact tables: Fact tables are used to hold numerical and quantifiable data, often known as metrics or measurements. They include the major data points being studied, such as sales income, quantities sold, and client orders. Fact tables are often connected to dimension tables via foreign key connections.

6.5.1 Query Execution in ROLAP:

ROLAP systems use SQL queries to retrieve and alter data from relational databases. Query execution in ROLAP consists of the following steps:

- **Query Parsing:** The ROLAP engine parses the SQL query entered by the user or application, identifying the tables and columns referred to in the query.
- **Query Optimization:** The ROLAP engine optimizes the query execution plan to reduce the time and resources necessary to get the desired data. This optimization process may employ a variety of strategies, including index selection, join optimization, and query rewriting.
- **Data Retrieval:** The ROLAP engine runs the optimized SQL query against the relational database to get relevant data from the dimension and fact tables.
- **Data Aggregation:** If the query requires aggregations or summarizations, such as computing totals, averages, or counts, the ROLAP engine applies SQL aggregate functions (e.g., SUM, AVG, COUNT) to the received data.
- **Result Presentation:** Finally, the ROLAP engine displays the query results to the user or application in a tabular manner or via visualization tools, allowing for additional analysis and interpretation.

Note: ROLAP systems store data in relational databases with tables representing dimensions and facts. SQL queries are utilized for query execution, and the ROLAP engine parses, optimizes, and retrieves data from the relational database based on the user's query specifications. This strategy enables ROLAP systems to benefit from the flexibility and scalability of relational databases while still allowing users to do complicated data analysis using familiar SQL terminology.

6.6 Data Storage in MOLAP

Multidimensional OLAP systems store data in specialized forms known as multidimensional cubes. These cubes organize data in various dimensions, resulting in a multidimensional representation of the data. This is how data storage works in MOLAP.

1. **Multidimensional cubes:**
    - MOLAP systems arrange data into multidimensional cubes, with each cell representing a data point intersecting many dimensions.
    - Dimensions reflect various data elements or qualities, such as time, region, product, and customer.
    - Each dimension is made up of hierarchies that arrange data at varying levels of granularity. For example, a time dimension may include hierarchies for year, quarter, month, and day.

2. **Data Aggregation:**
    - MOLAP cubes usually pre-aggregate data at various degrees of granularity to increase query speed.
    - Aggregated data is kept in the cube alongside detailed data, making it possible to quickly obtain summary information without having to compute aggregations on the fly.

3. **Indexed Storage**
    - MOLAP cubes frequently employ unique indexing techniques to improve data storage and retrieval.
    - These indexes enable efficient access to data based on various combinations of dimensions, allowing for quick query processing.

6.6.1 Query Processing in MOLAP

MOLAP systems are designed for multidimensional query processing, allowing users to examine data from many angles. Query processing in MOLAP operates as follows:

1. **Query parsing**
    - The MOLAP engine parses user queries to determine the dimensions, hierarchies, and measurements specified in the query.
2. **Cube Navigation**
    - The MOLAP engine navigates the multidimensional cube to find relevant data based on the dimensions supplied in the query.
    - It employs efficient indexing and storing mechanisms to enable rapid access to the necessary data points within the cube.
3. **Data retrieval**
    - The MOLAP engine extracts the desired data from the cube, including detailed and aggregated information.
    - Higher-level summaries yield aggregated data, whereas precise intersections of dimensions provide detailed data.
4. **Calculation and analysis:**
    - After the data is obtained, the MOLAP engine may conduct the computations, aggregations, and other analytical operations indicated in the query.
    - Users may discover insights by analyzing data with interactive tools and visualizations, exploring multiple dimensions and hierarchies.

Notes: MOLAP systems store data in multidimensional cubes, allowing for efficient storage and retrieval across several dimensions. These cubes are pre-aggregated and indexed to improve query efficiency, allowing for rapid and interactive multidimensional analysis. MOLAP engines browse cubes to gather desired data, do computations and analyses, and provide the results to users for further investigation and decision-making.

6.7 Data warehouse backend process

Several critical elements in the data warehouse backend process guarantee that data is maintained efficiently and effectively for analytical insights and decision-making. The process often begins with data extraction, which involves gathering information from multiple operating systems, external sources, or other data repositories. Extraction entails finding relevant data sets using preset criteria or constraints. Once retrieved, the data goes through a number of transformations before being stored and analyzed in the data warehouse.

Transformation is a vital process in which the extracted data is cleaned, filtered, and structured to match with the data warehouse's schema and standards. This process ensures that data is consistent, accurate, and usable by resolving inconsistencies, addressing missing values, and implementing business rules or transformations. Data can also be enhanced with extra features or derived metrics to increase its analytical usefulness.

Following transformation, the data is put into the data warehouse using a process known as ETL (Extract, Transform, Load). ETL is the process of entering converted data into the relevant tables or structures inside the warehouse, usually through batch processing or incremental updates. Loading may involve data validation procedures to guarantee integrity and completeness prior to integration into the warehouse.

Once data is successfully imported into the warehouse, continual maintenance is required to ensure data quality, consistency, and performance. Maintenance duties may involve removing duplicates or outdated records on a regular basis, improving data storage and indexing for efficient retrieval, and monitoring system performance to discover and resolve bottlenecks or difficulties.

6.8 Strategies for efficient data storage

Efficient data storage solutions, such as partitioning and indexing, are critical for improving performance in data warehouses. Partitioning divides huge tables into smaller pieces, which improves query efficiency and simplifies data administration responsibilities. Indexing builds data structures that enable fast data retrieval, improve query response times, and satisfy data integrity restrictions. Organizations that employ these tactics successfully can achieve quicker query processing, more scalability, and overall efficiency in their data warehouse settings. Furthermore, these strategies allow data warehouses to manage higher amounts of data and increase workloads while retaining excellent performance and dependability.

6.9 Practical applications of OLAP and data warehousing in business intelligence

OLAP and data warehousing are critical components of modern corporate intelligence, enabling a diverse set of analytical applications. These technologies enable enterprises to get meaningful insights from massive volumes of data, resulting in more informed decision-making and strategic planning.

OLAP and data warehousing have practical uses, such as reporting, which allows for the generation of customisable reports summarizing critical business KPIs. Furthermore, they provide dynamic dashboards that provide real-time insights into corporate performance, allowing users to easily monitor KPIs and track progress.

Trend analysis is another critical application that benefits from OLAP and data warehousing's ability to aggregate historical data and find trends. This allows firms to better understand historical trends and predict future results. Furthermore, OLAP and data warehousing aid forecasting efforts by utilizing predictive modeling approaches to anticipate market trends and client behavior.

Furthermore, these technologies enable users to perform ad hoc analysis, giving them the freedom to explore data interactively and discover hidden insights. Overall, OLAP and data warehousing provide the foundation of business intelligence systems, providing the required infrastructure and analytical skills to propel corporate success.

6.10 Scalability of OLAP engines and data warehousing systems

Scalability of OLAP engines and data warehousing systems is measured by their capacity to manage rising data quantities and user demands while preserving performance and dependability. Scalability is influenced by a variety of factors including as design, hardware resources, data distribution, and query optimization strategies. OLAP engines and data warehouse systems may use multiple scalability methods, such as vertical or horizontal scaling. When evaluating scalability, it is critical to consider data distribution, query optimization, concurrency management, and resource allocation. Also, examine the system's adaptability and extensibility to adapt to changing data volumes and user demands over time.

Scalability testing, performance benchmarking, and capacity planning are important elements in examining the scalability of OLAP engines and data warehouse systems to guarantee they can satisfy growing data volumes and user demands.

Check Your Progress

1. How does ROLAP differ from MOLAP in terms of data storage and query execution?
2. What are some key advantages of ROLAP systems in handling large datasets and dynamic data environments?
3. Can you provide an example of a use case where ROLAP excels in providing complex queries and ad-hoc analysis?
4. How does ROLAP support data integration from multiple sources, and what benefits does this feature offer to organizations?

5. In what industries or business functions is ROLAP commonly used for compliance reporting, and what are the specific challenges it helps address?
6. Explain the role of dimension tables and fact tables in the data storage architecture of ROLAP systems, and how are they utilized in facilitating multidimensional analysis?
7. How does the storage architecture of MOLAP differ from that of ROLAP, and what advantages does the multidimensional cube structure offer in terms of data

6.11 Summing Up

- **OLAP Engine**: OLAP engines are specialized software components that enable multidimensional data processing. They let users to examine and evaluate data from a variety of viewpoints, including time, region, and product categories, in order to get insights and make educated decisions.

- The data warehouse backend process consists of various steps, including data extraction, transformation, loading (ETL), and maintenance. Data is taken from many sources, processed to meet the warehouse schema, fed into the data warehouse, and managed to assure data quality, consistency, and long-term performance.

- ROLAP vs. MOLAP: OLAP systems are classified as either Relational OLAP (ROLAP) or Multidimensional OLAP (MOLAP). ROLAP systems store data in relational databases and utilize SQL queries for analysis, whereas

MOLAP systems use multidimensional structures such as cubes to improve query performance.

- OLAP engines and data warehouses are commonly used in business intelligence and analytics applications to perform activities like as reporting, dash boarding, trend analysis, forecasting, and ad-hoc analysis. They lay the groundwork for deriving useful insights from massive amounts of data to aid in decision-making processes.

- Performance and scalability are important factors for OLAP engines and data warehouses. Indexing, segmentation, and query optimization are all performance improvement strategies. Scalability is also required to support growing data quantities and user demands, with possibilities for vertical or horizontal scaling depending on the system design.

6.12 Answer to Check your Progress

1. ROLAP saves data in relational databases and retrieves it using SQL queries, making it adaptable and ideal for complicated, ad hoc searches and big datasets. MOLAP, on the other hand, stores data in multidimensional cubes and pre-aggregates it to improve query performance. This method is best suited for recurring queries and predetermined analyses, providing improved performance for activities such as trend analysis and forecasting.

2. ROLAP systems manage massive datasets and dynamic data environments effectively by employing scalable relational databases and supporting complicated, ad-hoc searches with SQL. They give flexibility in data analysis and interact smoothly with current database systems. This assures quick

data updates and real-time analytical capabilities, making ROLAP excellent for applications with rapid data changes and broad, flexible querying requirements.

3. A healthcare company that uses ROLAP excels at assessing patient information, treatment results, and operational indicators. With continuously changing data, ROLAP offers complicated searches to uncover patterns, analyze therapy success, and optimize patient care. The technology enables healthcare analysts to execute ad hoc analysis on massive datasets, such as analyzing the impact of new medicines or comparing outcomes across diverse patient demographics, giving deep, flexible insights critical for decision-making.

4. ROLAP makes data integration from several sources easier by easily integrating with relational databases and external repositories. This skill allows firms to aggregate data for extensive analysis, resulting in better decision-making processes. By offering a single picture of processes, ROLAP minimizes duplication, assures data consistency, and enables cross-functional analysis. This function improves organizational efficiency and promotes informed decision-making by providing greater insights into business performance.

5. ROLAP is commonly used in finance, healthcare, and pharmaceuticals for compliance reporting. It handles enormous transaction quantities, assures precise patient data analysis, and monitors medication trials and manufacturing operations. ROLAP's ability to handle sophisticated queries and integrate several data sources tackles issues such as data accuracy, timeliness, and thorough analysis, making it critical for achieving severe regulatory standards in these industries.

6. Dimension tables include descriptive features that offer context for the data, such as time, region, and product categories. Fact tables include numerical data like as sales income or quantities sold, and they are connected to dimension tables via foreign key connections. Dimension and fact tables serve as the foundation for multidimensional

analysis in ROLAP systems, allowing users to examine data across several dimensions and monitor business success.

7. MOLAP stores data in multidimensional cubes, whereas ROLAP uses relational databases. MOLAP's multidimensional cube structure provides benefits such as quicker data retrieval and query speed thanks to pre-aggregated data and optimal storage. MOLAP allows queries to obtain summarized data straight from the cube, minimizing the need for complicated joins and calculations, leading in faster response times and better overall performance than ROLAP systems.

8. Hierarchies in MOLAP arrange data into logical frameworks, which allows for multidimensional analysis. Aggregation summarizes data at various levels of these hierarchies, which improves query performance. Hierarchies enable people to browse and evaluate data at different degrees of detail. This technique improves efficiency in multidimensional analysis by reducing data exploration and allowing for rapid insights from summarized data, resulting in better decision-making processes inside businesses.

9. OLAP and data warehousing have practical applications across sectors, including business intelligence activities such as reporting, dashboards, trend analysis, and forecasting. They enable ad hoc analysis for interactive data exploration and insights discovery. Furthermore, these technologies are used in compliance reporting for industries like as banking, healthcare, and pharmaceuticals to ensure regulatory compliance and data integrity. Overall, OLAP and data warehousing help firms make data-driven decisions and plan strategically.

6.13 Possible Questions

Multiple Choice Questions:

1. What is the primary purpose of Online Analytical Processing (OLAP)?

a) Handling transactional data
b) Studying multidimensional data
c) Performing data integration
d) Supporting real-time processing

2. What types of queries does OLAP excel at handling?

a) Data entry queries
b) Complex computations
c) Basic queries
d) Transactional queries

3. Which of the following is not a form of OLAP?
a) ROLAP
b) MOLAP
c) DOLAP
d) HOLAP

4. Which OLAP form combines elements of both ROLAP and MOLAP?
a) ROLAP
b) MOLAP
c) HOLAP
d) DOLAP

5. Which of the following is the initial step in the data warehouse backend process?
a) Data transformation
b) Data extraction
c) Data analysis
d) Data loading

Answer: 1. (b), 2 (a), 3. (c), 4. (c), 5. (b)

Fill in the Following Blanks:

1. The process often begins with data ________, which involves gathering information from multiple operating systems, external sources, or other data repositories.

2. Transformation is a vital process in which the extracted data is cleaned, filtered, and structured to match with the data warehouse's ________ and standards.

3. In the field of business intelligence and data management, OLAP engines support __________ analysis, allowing users to interact with and understand data from multiple perspectives.

4. Data warehouses' backend procedures include data extraction, transformation, loading, and __________, which all work together to ensure that data is accurate, integrated, and easy to analyze.

5. Data warehouses function as centralized stores for large amounts of __________ data, assuring its availability and integrity.

Answer: 1.Extraction 2.Schema 3.Multidimensional 4.Management 5.Structured

State Whether True or False

1. OLAP engines support multidimensional analysis, allowing users to interact with and understand data from multiple perspectives.
2. Data warehouses function as decentralized stores for large amounts of structured data.
3. Data warehouse backend procedures include data extraction, transformation, loading, and management, ensuring data accuracy and integration.

4. ROLAP engines utilize multidimensional cubes for data storage and analysis.

5. MOLAP engines pre-aggregate data for faster query performance compared to ROLAP engines.

Answer: 1.True 2.False 3.True 4.False 5.True

Short Answer Questions

1. What are the primary functions of OLAP engines in business intelligence?
2. What are the key components of data warehouse backend procedures?
3. How do ROLAP and MOLAP engines differ in their approach to data analysis?
4. What are the practical applications of OLAP engines and data warehouses in improving company operations?
5. How do OLAP engines and data warehouses contribute to enhancing data-driven decision-making within organizations?

Long Answer Questions

1. Describe the key differences between ROLAP and MOLAP engines. How do these differences impact their suitability for different types of business intelligence applications?

2. Explain the comprehensive backend processes involved in data warehousing. Discuss the steps of dataextraction, transformation, loading (ETL), and data management.

3. Discuss the practical applications of OLAP engines and data warehouses in business intelligence. Explain how these technologies enhance decision-making and operational efficiency.

4. How do OLAP engines support multidimensional analysis, and why is this capability critical for strategic decision-making?

5. Assess the role of data warehouses as centralized stores for large amounts of structured data.What challenges do organizations face in managing such data.

---×---

BLOCK- II

UNIT: 1
ASSOCIATION RULES

Unit Structure:

1.1 Introduction

In this unit we will learn about the problem of deriving associations from data which have received a great deal of attention. In 1993, Agrawal et al formulated a problem which is known as the market basket problem. In this problem a large set of items and a basket of these items is given. The basket is considered to be the subset of a large collection of transactions of these items. Now the task is to find relationships between the presences of various items within these baskets. This market basket problem finds a number of applications in data mining. One popular example is the supermarket. The supermarkets analyses the customer's buying patterns and forms strategies for their better revenue by giving offers and discounts on particular items. This unit will start with explanation about the market basket problem and will give a good understanding about the basic terms, ideas and concepts related to association rules.

1.2 Unit Objectives

After going through this unit, you will be able to

- Understand the market basket problem
- Define the basic terms such as support, confidence of an itemset
- Define the association rule, frequent set, maximal frequent set and border set.
- Learn the methods to discover association rules
- Learn the properties of the frequent sets.

1.3 Market Basket Analysis

Market Basket Analysis: Market Basket Analysis is a technique which is used in retail setting to analyze the purchasing pattern of the customer and thus to create a link between the products that are frequently bought together by the customers. It helps the retailers to keep items available in their inventory based on the customer's choice. The retailer may also offer certain offers to a product which might be kept beside the product which is most likely purchased by the customer. In doing so a customer buying a product may get inclined to the product offering discount and hence may end up buying the two products. Market Basket Analysis helps in creating strategies for better sale of the products which is a huge advantage for the retailers. Data mining techniques used in Market Basket Analysis helps in finding out the patterns and the hidden relationships between the customer's buying pattern. The source for discovering these patterns are the vast amount of data that are accumulated in any retailing system and hence collected and stored for future benefits.It considerably helps in decision-making processes related to cross-marketing, catalogue design, and consumer shopping analytics. Market Basket Analysis mainly works following the rule:

IF {} THEN {}

For example if a certain customer buys a shampoo then it is evident that the customer might buy the conditioner also. Hence in this case the rule can be implemented as:

IF (shampoo) THEN (conditioner)

The associated products can be kept together so that they are purchased also together and hence it benefits the retailers with better revenue.

There are terminologies that are used in Market Basket Analysis:

Itemset: A set of items that occur together in a transaction set is called an itemset. In other words these are the items that are purchased together by a customer. For example if we apply the rule mentioned earlier then IF (BREAD, MAYONNAISE) THEN (OREGANO). It may also be possible that are no items in the itemset as some of the items are ignored by the other items in the dataset.

Support count: Support count is the frequency or number of times an itemset appears in a transaction database.Support count can also be stated as probability. For instance, if bread has a support count of 40 out of a possible 200 transactions, then the probability or the support count is 40/200 or 0.2.

Confidence: Confidence is the conditional probability of the association of items that are being purchased together.

Antecedent:The IF component written on the left-hand side or the item sets within the data are referred to as antecedents.

Consequent:The THEN component or an item or itemset found in combination with the antecedent is called the consequent.

The concept of support and confidence are discussed more elaborately in the next section.

Check Your Progress

1. What is the main idea behind Market Basket Analysis?
2. What is an itemset?
3. List two applications of Market Basket analysis.

1.4 Association Rule

Definition: Association Rule

For a given transaction database T, an association rule is an expression of the form X=>Y, where X and Y are subsets of A and X=>Y holds with confidence τ, if τ% of transactions in D that supports X also support Y. The rule X =>Y has support σ in the transaction set T if σ% of transactions in T support X∪Y.

The intuitive meaning of such a rule is that a transaction of the database which contains X tends to contain Y. Given a set of transactions, T, the problem of mining association rules is to discover all rules that have support and confidence greater than or equal to the user-specified minimum support and minimum confidence respectively.

The formal definitions of support and confidence are:

Support, s(X=>Y) = σ(X∪Y) ,where N is the total number of transactions.

N

Confidence, c(X=>Y) = σ(X∪Y)

σ(X)

Example: Consider the previous figure. Assume that σ =50% and τ = 60%. Clearly the rule Milk=> Cheese holds. The confidence of this rule is, in fact 100% because all the transactions that supports Milk also support Cheese. But on the other hand the rule Cheese=> Milk also holds but its confidence is 60% only.

STOP TO CONSIDER

Association rules can be used extensively in diagnosis of disease. As diagnosis of a particular disease is both important and difficult, association rules might play a very important rule in finding out associations between the probability of the disease and it's symptoms. It will be a great benefit for the medical experts.

Self Assessment Question

Define support and confidence by taking an example of your own.

1.4.1 Methods to discover Association Rules

The association rule mining problem can be formally stated as follows:

Given a set of transactions T, find all the rules having support $\geq$minsup and confidence$\geq$minconf, where minsup and minconf are the corresponding support and confidence thresholds.

A common strategy adopted by many association rule mining algorithms is to decompose the problem into two major subtasks:

1. **Frequent Itemset Generation**, whose objective is to find all the itemsets that satisfy the minup threshold. These itemsets are called frequent itemsets.

2. **Rule Generation**, whose objective is to extract all the high confidence rules from the frequent itemsets found in the previous step.

1.4.2 Frequent Set and Border Set

Definition: Frequent Set

Let T be the transaction database and σ be the user specified minimum support. An itemset $X \subseteq A$ is said to be a frequent itemset in T with respect to σ, if

$$s(X)_T \geq \sigma$$

In the previous example if σ = 50% is assumed, then the set {Milk, Cheese} is a frequent set as it is supported by at least 3 out of 5 transactions.

Definition:Maximal Frequent Set

A frequent set is a maximal frequent set if it is a frequent set and no superset of this is a frequent set.

Definition: Border Set

An itemset is a border set if it is not a frequent set, but all its proper subsets are frequent sets**.**

1.3.3 Properties of Frequent Sets

The set of frequent sets for a given T, with respect to σ exhibits some interesting properties.

- **Downward Closure Property:** Any subset of a frequent set is a frequent set.
- **Upward Closure Property:** Any superset of an infrequent set is an infrequent set.

1.4.4 Mining Frequent Sets

Mining frequent sets is to find frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories. Let's understand in details what exactly is mining frequent sets

Given a set of items $I = \{i_1,i_2,i_{3,\ldots\ldots}i_m\}$ and a database of transactions D, where a transaction $T \subseteq I$ is a set of items there are two tasks in mining frequent sets:

1. Find all the subsets of items that occur together in more than one transaction.

2. Generate all the rules that associate the presence of one set of items with another set of items in the transaction database.

How these tasks are accomplished using the different association rule algorithm techniques will be discussed in the next chapter in details.

1.5 Summing Up

1. In 1993, Agrawal et al formulated a problem which is known as the market basket problem.

2. Market Basket Analysis is a technique which is used in retail setting to analyze the purchasing pattern of the customer and thus to

create a link between the products that are frequently bought together by the customers

3. It considerably helps in decision-making processes related to cross-marketing, catalog design, and consumer shopping analytics.

4. Theterminologies that are used in Market Basket Analysis are: itemset, support count, confidence, antecedent, consequent.

5. Given a set of transactions, T, the problem of mining association rules is to discover all rules that have support and confidence greater than or equal to the user-specified minimum support and minimum confidence respectively.

6. Association rule mining algorithm mainly have two major subtasks:

a) Frequent itemset generation

b) Rule generation

7. Let T be the transaction database and σ be the user specified minimum support. An itemset $X \subseteq A$ is said to be a frequent itemset in T with respect to σ, if

$$s(X)_T \geq \sigma$$

8. A frequent set is a maximal frequent set if it is a frequent set and no superset of this is a frequent set.

9. An itemset is a border set if it is not a frequent set, but all its proper subsets are frequent sets**.**

10. Two important properties of frequent set are: Upward and Downward Closure Property.

1.6 Answers to Check Your Progress

1. The main idea behind market basket analysis is to find out associations between customer’s purchasing pattern and thus to create a relationship between the products they frequently purchase together.

2. A set of items that occur together in a transaction set is called an itemset.

3. Market Basket analysis helps in decision-making processes related to cross-marketing, catalogue design, and consumer shopping analytics.

1.7 Possible Questions

1. Explain the market basket analysis problem with the help of an example.

2. What is an association rule?

3. Define support and confidence of an itemset.

4. What are the two methods to discover association rules?

5. Define frequent set, maximal frequent set and a border set?

6. What do you mean by mining frequent sets? Explain

1.8 References and Suggested Readings

1. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

2. Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

3. Chaudhary S. *Market Basket Analysis: Anticipating Customer Behavior* https://www.turing.com/kb/market-basket-analysis

---×---

UNIT: 2
ASSOCIATION RULE ALGORITHMS I

Unit Structure:

2.1 Introduction

As now we have a very clear understanding of market basket data, frequent sets, the downward and the upward closure property, we can learn about the algorithms for mining the association rules. We can accept the fact that for the discovery of association rules, discovery of frequent sets is essential. In this unit we will study about the different methods of discovering association rules and the various tasks associated with it. Apriori algorithm which is the most popular algorithm for the discovery of frequent sets and hence important for mining the association rules will be discussed in details. We will also learn about three more algorithms which are namely: Pincer Search algorithm, Incremental algorithm and Border algorithm. All these methods differ in the way it handles the candidate sets and the method of reducing the number of database passes.

2.2 Unit Objectives

After going through this unit, you will able to:

- Understand the Apriori algorithm
- Understand the Pincer search algorithm
- Understand the Incremental algorithm
- Understand the Border algorithm

2.3 Algorithm for Mining Association Rules

There are different algorithms for mining association rules. We are going to start with the most popular one i.e the Apriori algorithm. After understanding it, we will go through three other algorithms Pincer search algorithm, Incremental algorithm and Border algorithm.

2.3.1 *A-priori* Algorithm

It is also called the level-wise algorithm. It was proposed by Agrawal and Srikant in 1994. It is the most popular algorithm to find all the frequent sets. It makes use of the downward closure property. By convention, the algorithm assumes that items within a transaction or itemset are sorted in lexicographic order. It employs an iterative approach known as a level-wise search, where (k − 1)-itemsets are used to explore k itemsets.

First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L_1. Next, L_1 is used to find L_2, which is then used to find L_3, and so on, until no more frequent itemsets can be found. The finding of each L_k,requires one full scan of D. To improve the efficiency of the level-wise generation of frequent itemsets, the downward closure property is considered. This property is based on the following observation. If an itemset A does not satisfy the minimum support threshold, min sup, then A is not frequent; i.e. $P(A) < minsup$. If an item B is added to the itemset A, then the resulting itemset $A \cup B$

cannot occur more frequently than A. Therefore $A \cup B$ is not frequent either, that is $P(A \cup B) <$ minsup.

A two-step process is used to find L_k, from L_{k-1}, for $k \geq 2$:

1. Candidate Generation: To find L_k, a set of candidate (k−1) itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted by C_k. Let l_1 and l_2 be itemsets in L_{k-1}.

The notation l_i [j] refers to the jth item in l_i . Thus in l_1 , the last item and the next to the last item are given respectively by l_1 [k −1] and l_1 [k −2]. Any two itemsets L_k−1 are joined if their first (k −2) items are in common. The candidate generation method is described below:

gen_candidate_itemsets with the given L_{k-1} as follows:

```
$C_k = \emptyset$
for all itemsets $l_1 \in L_{k-1}$ do
for all itemsets $l_2 \in L_{k-1}$ do
        if $l_1[1] = l_2[1] \wedge l_1[2] = l_2[2] \wedge \ldots\ldots \wedge l_1[k-1] < l_2[k-1]$
        then c= $l_1[1], l_1[2], \ldots. l_1[k-1], l_2[k-1]$
         $C_k = C_k \cup c$
```

The condition l_1 [k − 1] <l_2 [k − 1] ensures that no duplicates are generated. The resulting itemset formed by joining l_1 and l_2 is { l_1 [1], l_1 [2], . . . , l_1 [k − 2], l_1 [k − 1], l_2 [k − 1]}.

2. The prune step: Set C_k is a superset of L_k, because although all the frequent k itemsets are included in C_k, its members may or may not be frequent. One could scan the database to determine the count of each candidate in C_k and eliminate any itemset that does not meet the minimum support threshold.This would then give L_k. However, C_k can be huge, and so this could be very time-consuming.

To eliminate the infrequent itemsets, the Apriori property is used as follows. Any (k−1) itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence, if any (k−1)itemset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot

be frequent either and so can be removed from C_k . The pruning algorithm is described below:

```
prune($C_k$)
for all $c \in C_k$
for all (k-1) subsets d of c do
        if $d \notin L_{k-1}$
        then $C_k = C_k \setminus \{c\}$
```

***A Priori* Algorithm**

```
Initialize: k:=1, $C_1$ = all the 1-itemsets;
Read the database to count the support of $C_1$ to determine $L_1$
$L_1$:= {frequent 1-itemsets};
k:= 2; // k presents the pass number//
while($L_{k-1} \neq \emptyset$) do
begin
$C_k$ := gen_candidate_itemsets with the given $L_{k-1}$
prune($C_k$)
for all transactions $t \in T$ do
increment the count of all candidates in $C_k$ that are contained in t;
$L_k$ := All candidates in $C_k$ with minimum support;
k:=k+1
end

Answer:= $\cup_k L_k$
```

Example: Consider a database, D, consisting of 9 transactions.

TID	List of Items
T1	1, 2, 5

T2	2, 4
T3	2, 3,6
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

Suppose min_support count required is 2 (i.e min_sup = 2/9 = 22%). Let minimum support required is 70%.

In the first iteration of the algorithm, each item is a member of the set of candidates 1-itemsets, C_1. The algorithm simply scans all the transactions in order to count the number of occurrences of each item.

Step 1: Generating 1-itemset frequent pattern

C_1 Itemset	Support Count
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2
{6}	1

The set of frequent 1-itemsets, L1, consists of the candidate itemsets satisfying the minimum support count of 2. Thus all the candidates in C1, except for {6}, are in L1.

L_1 Itemset	Support Count

{1}	6
{2}	7
{3}	6
{4}	2
{5}	2

To discover the set of frequent 2-itemsets, L_2, the algorithm joinsL_1 with itself to generate a candidate set of 2-itemsets, C_2. Note that no candidates are removed from C_2 during the pruning step since each subset of the candidates is also frequent.

C2 itemset
{1, 2}
{1, 3}
{1, 4}
{1, 5}
{2, 3}
{2, 4}
{2, 5}
{3, 4}
{3, 5}
{4, 5}

Next the transactions in D are scanned and the support count of each candidate itemset in C_2 is accumulated.

L_2 Itemset	Support Count
{1,2}	4
{1,3}	4
{1,5}	2
{2,3}	4

{2,4}	2
{2,5}	2

Next, C_3 is generated by joining L_2 with itself.

The result is C_3 = {{1, 2, 3}, {1, 2, 5}, {1, 3, 5}, {2, 3, 4}, {2, 3, 5}, {2, 4, 5}}.

C_3 is then pruned using the Apriori property: All nonempty subsets of a frequent itemset must also be frequent. From the way each candidate of C3 is formed, it is clear that is needed is to check, is the subset obtained from the last two members of the candidate set.

Since {2, 3} is a frequent itemset, {1, 2, 3} in C_3 is kept

Since {2, 5} is a frequent itemset, {1, 2, 5} in C_3 is kept

Since {3, 5} is not a frequent itemset, {1, 3, 5} is removed from C_3.

Since {3, 4} is not a frequent itemset, {2, 3, 4} is removed from C_3.

Since {3, 5} is not a frequent itemset, {2, 3, 5} is removed from C_3.

Since {4, 5} is not a frequent itemset, {2, 4, 5} is removed from C3.

Therefore after pruning, C_3 is given by:

C_3 itemset
{1, 2, 3}
{1, 2, 5}

The transactions in D are scanned to determine L_3, consisting of those candidates 3-itemsets in C_3 having at least minimum support.

C_3 itemset	Support Count
{1,2,3}	2
{1,2,5}	2

Since both 3-itemsets in C_3 have the least minimum support, L_3 is therefore given by:

L_3 itemset	Support Count
{1,2,3}	2
{1,2,5}	2

Finally L_3 is joined with itself to generate a candidate set of 4-itemsets, C_4. This results in a single itemset {1, 2, 3, 5}. However this itemset is pruned since its subset {3, 5} is not frequent. Thus, $C_4 = \varnothing$, and the algorithm terminates, having found all of the frequent itemsets.

Check Your Progress

1. What is the purpose of prune step in Apriori algorithm?
2. Which property used is used in Apriori algorithm?

2.3.2 Pincer Search Algorithm

Pincer Search Algorithm was proposed by, Dao-I Lin & Zvi M. Kedem of New York University in 1997. This algorithm uses both, the top-down and bottom-up approaches to Association Rule mining. It is a slight modification to original Apriori Algorithm by R. Aggarwal & Srikant. In this the main search direction is bottom-up (same as Apriori) except that it conducts simultaneously a restricted top-down search, which basically is used to maintain another data structure called Maximum Frequent Candidate Set (MFCS). As output it produces the Maximum Frequent Set i.e. the set containing all maximal frequent itemsets, which therefore specifies immediately all frequent itemsets. The algorithm specializes in dealing with maximal frequent itemsets of large length.

Pincer Search combines the following two approaches:

Bottom-up: Generate subsets and go on to generating parent-set candidate sets using frequent subsets.

Top-Down: Generating parent set and then generating subsets.

It also uses two special properties: **Downward Closure Property** and **Upward Closure Property.**

It uses the above two properties for pruning candidate sets, hence decreases the computation time considerably. It uses both approaches for pruning in following way:

If some maximal frequent item set is found in the top down direction, then this item set can be used to eliminate (possibly many) candidates in the bottom-up direction. The subsets of this frequent item set will be frequent and hence can be pruned (acc. to Downward closure property).

If an infrequent item set is found in the bottom up direction, then this infrequent item set can be used to eliminate the candidates found in top-down search found so far (acc. to Upward Closure Property).

This two-way approach makes use of both the properties and speeds up the search for maximum frequent set. The algorithm begins with generating 1-itemsets as Apriori algorithm but uses top-down search to prune candidates produced in each pass. This is done with the help of MFCS set.

Let MFS denote set of Maximal Frequent sets storing all maximally frequent item sets found during the execution. So at anytime during the execution MFCS is a superset of MFS. Algorithm terminates when MFCS equals MFS.

In each pass over database, in addition to counting support counts of candidates in bottom-up direction, the algorithm also counts supports of item sets in MFCS: this set is adapted for top-down search.

Consider now some pass k, during which item sets of size k are to be classified. If some item set that is an element of MFCS, say X of cardinality greater than k is found to be frequent in this pass, then all its subsets will be frequent. Then all its subsets of cardinality k are pruned from the set of candidates considered in bottom-up approach in this pass. They and their supersets will never be candidates again. But, they are not forgotten and used in the end.

Similarly when a new item sets is found to infrequent in bottom-up direction, the algorithm makes use of it to update MFCS, so that no subset of any item set in MFCS should have this infrequent item set as its subset.

By use of MFCS maximal frequent sets can be found early in execution and hence improve performance drastically.

Note: In general, unlike the search in bottom-up direction, which goes up one level in one pass, *the top down search can go down many levels in one pass.*

Pincer Search Method

```
$L_0 = \varnothing$ ; k=1; $C_1 = \{\{i\} \mid i \in I\}$; $S_0 = \varnothing$
MFCS = {{ 1,2,…, n}}; MFS = $\varnothing$ ;

do until $C_k = \varnothing$ and $S_{k-1} = \varnothing$
        read the database and count support for $C_k$& MFCS.
MFS = MFS $\cup$ {frequent itemsets in MFCS};
$S_k$ = {infrequent itemsets in $C_k$ };

call MFCS_gen algorithm if $S_k \neq \varnothing$;
call MFS_pruning procedure;
generate candidates $C_{k+1}$ from $C_k$ ;
if any frequent itemset in $C_k$ is removed from MFS_pruning
procedure
callrecovery procedure to recover candidates to $C_{k+1}$ .
call MFCS_prune procedure to prune candidates in $C_{k+1}$ .
k=k+1;
return MFS

MFCS_gen
for all itemsets $s \in S_k$
        for all itemsets $m \in$ MFCS
        if s is a subset of m
```

```
          MFCS = MFCS \ {m}
          for all items e ∈ itemset s
                    if m \ {e} is not a subset of any itemset in MFCS
                    MFCS = MFCS ∪ {m \ {e}}
return MFCS
```

```
Recovery
for all itemsets l ∈ $L_k$
          for all itemsets m ∈ MFS
                    if the first k-1 items in l are also in m
                              for i from j+1 to |m|
                              $C_{k+1} = C_{k+1} \cup \{\{l.item_1, \ldots, l.item_k, m.item_i\}\}$
```

```
MFS_prune
for all itemsets c in $L_k$
if c is a subset of any itemset in the current MFS
delete c from $L_k$.
```

```
MFCS_prune
for all itemsets in $C_{k+1}$
if c is not a subset of any itemset in the current MFCS
delete c from $C_{k+1}$;
```

MFCS is initialized to contain one itemset, which contains all of the database items. MFCS is updated whenever new infrequent itemsets are found. If an itemset is found to be frequent then its subsets will not participate in subsequent support counting and candidate generation steps. If some itemsets in L_k are removed, the algorithm will call the *recovery* procedure to recover missing candidates.

Example: Customer Basket Database

Transaction	*Products*
1	Burger, Coke, Juice
2	Juice, Potato Chips
3	Coke, Burger
4	Juice, Groundnuts
5	Coke, Groundnuts

Step 1: $L_0 = \varnothing$, k=1;

C_1 = {{ Burger}, {Coke}, {Juice}, {Potato Chips}, {Ground Nuts}}

MFCS = {Burger, Coke, Juice, Potato Chips, Ground Nuts}

MFS = $\varnothing$;

Pass 1: Database is read to count support as follows:

{Burger}$\rightarrow$2, {Coke}$\rightarrow$3, {Juice}$\rightarrow$ 3, {Potato Chips}$\rightarrow$1, {Ground Nuts}$\rightarrow$2

{Burger, Coke, Juice, Potato Chips, Ground Nuts} $\rightarrow$ 0

So MFCS = {Burger, Coke, Juice, Potato Chips, Ground Nuts}

& MFS = $\varnothing$;

L_1 = {{Burger}, {Coke}, {Juice}, {Potato Chips}, {Ground Nuts}}

$S_1 = \varnothing$

At the moment since $S_1 = \varnothing$ there is no need of updating MFCS

C_2 = {{Burger, Coke}, {Burger, Juice}, {Burger, Potato Chips}, {Burger, Ground Nuts}, {Coke, Juice}, {Coke, Potato Chips}, {Coke, Ground Nuts}, {Juice, Potato Chips}, {Juice, Ground Nuts}, {Potato Chips, Ground Nuts}}

Pass 2: Read database to count support of elements in C_2& MFCS as given below:

{Burger, Coke} $\rightarrow$ 2, {Burger, Juice}$\rightarrow$1,

{Burger, Potato Chips}$\rightarrow$0, {Burger, Ground Nuts}$\rightarrow$0,

{Coke, Juice}$\rightarrow$1, {Coke, Potato Chips}$\rightarrow$0,

{Coke, Ground Nuts}$\rightarrow$1, {Juice, Potato Chips}$\rightarrow$1,

{Juice, Ground Nuts}$\rightarrow$1, {Potato Chips, Ground Nuts}$\rightarrow$0

{Burger, Coke, Juice, Potato Chips, Ground Nuts}$\rightarrow$0

MFS = $\varnothing$

L_2 = {{Burger, Coke}, {Burger, Juice}, {Coke, Juice}, {Coke, Ground Nuts}, {Juice, Ground Nuts}, {Juice, Potato Chips}}

S_2 = {{Burger, Potato Chips}, {Burger, Ground Nuts}, {Coke, Potato Chips}, {Potato Chips, Ground Nuts}}

For {Burger, Potato Chips} in S_2 and for {Burger, Coke, Juice, Potato Chips, Ground Nuts} in MFCS, new elements are acquired in MFCS as

{Burger, Coke , Juice, Ground Nuts} and {Coke, Juice, Potato Chips, Ground Nuts}

For {Burger, Ground Nuts} in S_2& for {Coke, Juice, Potato Chips, Ground Nuts} in MFCS, since {Burger, Ground Nuts} is not contained in this element of MFCS hence no action.

For {Burger, Coke, Juice, Ground Nuts} in MFCS two new elements are acquired

{Burger, Coke, Juice} & {Coke, Juice, Ground Nuts}

Since {Coke, Juice, Ground Nuts} is already contained in MFCS, it is excluded from MFCS

Now, MFCS = {{Burger, Coke, Juice}, {Coke, Juice, Potato Chips, Ground Nuts}}

Now, for {Coke, Potato Chips} in S_2 ,

MFCS = {{Burger, Coke, Juice}, {Coke, Juice, Ground Nuts}, {Potato Chips, Juice, Ground Nuts}}

Now, for {Potato Chips, Ground Nuts} in S_2,

MFCS = {{Burger, Coke, Juice}, {Coke, Juice, Ground Nuts}, {Potato Chips, Juice}}

Now, the candidate sets are generated as

C_3 = {{Burger, Coke, Juice}, {Potato Chips}}

Here algorithm stops since no more candidates need to be tested.

Then one scan is done for calculating actual support counts of all maximally frequent itemsets and their subsets. After this new rules are generated using minimum confidence threshold to get all *interesting* rules.

Check Your Progress

3. State the difference between the Apriori algorithm and Pincer search algorithm

2.3.3 Incremental Algorithm

In all the algorithms discussed earlier one common assumption is that the database itself does not change but in practical no transaction database is static. It is also possible to add more transactions intermittently to the database. Such type of changes in the database might potentially invalidate the existing set of association rules. These type of updates introduces many new association rules. So it is very important to update the association rules as soon as the database gets updated. The most important step in finding association rules is to discover the set of frequent sets. The set of frequent sets needs also to be recomputed as the database gets updated which means that the efforts made in computing the frequent sets before update becomes waste. If there is some intelligent way to make use of the earlier computations will be beneficial in reducing the amount of I/O operations.

Let us assume that T_{old} is the existing database for which the set of frequent itemsets L_{old} is already computed. The respective support values of the frequent sets in L_{old} are also known. Let us assume that a new set of transactions T_{new} is now added to the database to get $T_{whole} = T_{old} \cup T_{new}$. The problem of incremental computation of frequent sets is to determine the set of all frequent sets L_{whole} of T_{whole}.

Let us assume that L_{new} is the set of all frequent sets of T_{new}. Then the following observations are obvious:

1. Any itemset will be in L_{whole} if it is an element of both L_{new} and $L_{old.}$

2. Any itemset which is neither in L_{new} nor in L_{old} cannot be in $L_{whole.}$

Thus if we simply start finding L_{new}, we can use condition 1 given above to determine $L_{whole,}$ we are going to make some passes over $T_{new.}$ Hence we make use of these passes to compute the support of itemsets of L_{old} in T_{new}. That is, $s(X)_{Tnew}$ for X in $L_{old.}$ But unless we read T_{old}, we may not be able to find those frequent itemsets, if any, which are frequent in T_{new} as well as in T_{whole} but infrequent in T_{old}.

Keeping all the above observations in mind, let us design an incremental method. If we find $s(X)_{Tnew}$ for all X in L_{old} and if we can identify those sets which reach the σ level of support, , then we can partially characterize $L_{whole.}$ This is accomplished by making just one pass over T_{new} only. It is a partial characterization because this property takes into account the following cases:

a. The itemsets of L_{old} that are frequent in T_{new} can be determined.

b. The itemsets that belong to L_{old} but are not in L_{new} are automatically eliminated

c. The itemsets that are neither in L_{new} nor in L_{old} are not considered.

But those itemsets that are not in L_{old} but are frequent sets in T_{whole} are not taken into account in the above step.

A complete characterization can be achieved by finding L_{new} first and to compute the supports of the itemsets in $L_{new} \backslash L_{old}$ by making one pass over $T_{old.}$ This requires exactly one pass over the main database T_{old} and many passes over $T_{new.}$

This process may not turn out to be fruitful as no new frequent itemset may generate. Now the question is whether we can know in advance if any database pass is required or not. If there is no frequent set of T_{whole} which is not in L_{old}, then no additional database pass is required.

For determining this we need again a new concept called the Promoted Border Set.

Let X be an itemset which is not frequent in T_{old} , but is frequent in T_{new} as well as in T_{whole}. All the subsets of X are frequent in T_{whole} . Since X is not frequent in L_{old}, it has a subset that is a border set in $T_{old.}$ This border set becomes a frequent set after update.

An itemset that was a border set before update and become a frequent set after update is called a promoted border itemset. It is evident from the discussion above that $L_{whole} \cap (L_{new} \setminus L_{old}) \neq \emptyset$ if and only if there exists a promoted border set.

2.3.4 Border Algorithm

The Border algorithm maintains support counters for all frequent sets and all border sets. The algorithm works as follows. First the support counts of L_{old} and B_{old} are assumed to be known. The algorithm then starts by counting the support counts of the itemsets $L_{old} \cup B_{old}$ in T_{new}. For doing this, the algorithm requires to one pass over the increment portion of the database. It collects two categories of itemset F and B during this phase.

Set F contains the itemset of L_{old} which becomes a frequent set in $T_{whole.}$

Set B contains all the border sets whose support count reach the level σ and hence are promoted border sets. If there is no promoted border, then F contains all the frequent sets of $T_{whole.}$. If there is no promoted border, then F contains all the frequent sets of $T_{whole.}$But if there is at least one border set that becomes a promoted border, then the algorithm generates candidate sets which are supersets of the promoted border sets. It makes one pass over the database to count the support of these candidate sets. Thus the algorithm makes one pass over the increment database, and at most one pass over the whole database. The formal description of the border algorithm is given below.

Border Algorithm

```
read $T_{new}$ and increment the support count of X for all $X \in L_{old} \cup B_{old}$
F:= $\{X \mid X \in L_{old}$ and $s(X)_{Twhole} \geq \sigma \}$
B:= $\{X \mid X \in B_{old}$ and $s(X)_{Twhole} \geq \sigma \}$
Let m be the size of the largest element in B
***Candidate-generation***
***for all*** itemsers $l_1 \in B_{k-1} \cup C_{k-1}$ do begin
***for all*** itemsers $l_2 \in B_{k-1} \cup F_{k-1} \cup C_{k-1}$ do begin
```

```
        if $l_1[1] = l_1[1] \wedge l_1[2] = l_2[2] \wedge \ldots\ldots \wedge l_1[k-1] < l_2[k-1]$ then
        c = $l_1[1], l_1[2] \ldots.. l_1[k-1], l_2[k-1]$
        $C_k = C_k \cup \{c\}$
end do
end do

Prune $C_k$ for all k;
all subsets of k-1 size should be present in $B_{k-1} \cup F_{k-1} \cup C_{k-1}$
k:=k+1
Candidate C:= $\cup\ C_k$
read $T_{whole}$ and count the support values of each itemset in C.
new_frequent_sets:= $\{X \mid X \in C \text{ and } s(X)_{Twhole} \geq \sigma\}$
$L_{whole}$:= F $\cup$ new_frequent_sets
$B_{whole}$:= $(B_{old} \backslash B) \cup \{ X \in C$ and $s(X)_{Twhole} < \sigma$ and all its subsets are
in $L_{whole}\}$
```

Stop to Consider

The Borders algorithm is based on the notion of border sets, introduced in Mannila and Toivonen (1997).

Check Your Progress

4. What does presence of promoted border set indicate?

2.4 Summing Up

1. Apriori algorithm proposed by Agrawal and Srikant in 1994 also called as level wise algorithm is the most popular algorithm to find all the frequent sets.

2. It uses the downward closure property.

3.It employs an iterative approach known as a level-wise search, where (k − 1)-itemsets are used to explore k itemsets.

4. The two important steps in Apriori algorithm: Candidate generation and Prune step.

5. Pincer Search Algorithm was proposed by, Dao-I Lin & Zvi M. Kedem of New York University in 1997

6. This algorithm uses both, the top-down and bottom-up approaches to Association Rule mining.

7. In this the main search direction is bottom-up (same as Apriori) except that it conducts simultaneously a restricted top-down search, which basically is used to maintain another data structure called Maximum Frequent Candidate Set (MFCS).

8. It is very important to update the association rules as soon as the database gets updated.

9. The set of frequent sets needs also to be recomputed as the database gets updated which means that the efforts made in computing the frequent sets before update becomes waste.

10. An itemset that was a border set before update and become a frequent set after update is called a promoted border itemset.

2.5 Answers to Check Your Progress

1. The prune step in Apriori algorithm is used to eliminate infrequent itemsets that does not meet the minimum support threshold.

2. Apriori algorithm uses the downward closure property.

3. Pincer search algorithm is a slight modification to the original Apriori algorithm. In Pincer search algorithm the main search direction is bottom-up which is similar to Apriori but in addition it also makes a restricted top-down search which is basically used to maintain another data structure called Maximum Frequent Candidate Set.

4. An itemset that was a border set before update and becomes a frequent set after update is called a promoted border itemset.The existence of a promoted border set indicates that there is a need to read the old database once again.

2.6 Possible Questions

1. What are the two tasks carried out in Apriori algorithm?

2. Explain the Apriori algorithm with the help of an example.

3. State the working of Pincer search algorithm with the help of an example.

4. What is the advantage of incremental algorithm over other algorithms?

5. What is a promoted border set?

6. Explain the border algorithm with the help of an example.

2.7 References and Suggested Readings

1. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

2. Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

---×---

UNIT: 3
ASSOCIATION RULE ALGORITHMS II

Unit Structure:

3.1 Introduction

Most of the algorithms we have studied till now suffers from two shortcomings:

1) Unable to handle large number of candidate sets as it becomes very costly. For instance, if there are 10^4 frequent 1-itemsets, then approximately, 10^7 candidate 2-itemsets are generated. Moreover, if there is a frequent set of size 100, then roughly 10^{30}candidate sets are generated in this process.

2) Scanning the database repeatedly and check the large sets of candidates by pattern matching.

To overcome these shortcomings a new class of algorithm was developed called the Frequent Pattern Tree Growth algorithm. (FP-tree growth algorithm). In this unit we are going to study this algorithm in details along with some generalized association rules.

3.2 Unit Objectives

After going through this unit, you will

- Understand the FP tree growth algorithm
- Understand the generalized association rule
- Understand the association rule with item constraints

3.3 FP-tree growth Algorithm

FP-Tree growth algorithm is an efficient algorithm for producing the frequent itemsets without generation of candidate item sets. It is based upon the divide and conquer strategy. It needs two database scan for finding all frequent item sets. This approach compresses the database of frequent itemsets into frequent pattern tree recursively in the same order of magnitude as the number of frequent patterns, then in next step divides the compressed database into set of conditional databases.

Definition: FP-Tree

A frequent pattern tree (or FP-tree) is a tree structure consisting of an item-prefix tree and a frequent-item-header table.

- Item-prefix tree:
    - ➢ It consists of a root node labelled null.
    - ➢ Each non-root node consists of three fields
        - Item name
        - Support count and
        - Node link
- Frequent-item-header-table: It consists of two fields;
    - Item name
    - Head of node link which points to the first node in the FP-Tree carrying the item name

It may be noted that the FP-tree is dependent on the support threshold σ. For different values of σ, the trees are different. There is another typical feature of FP-tree; it depends on the ordering of the items. Different ordering may offer different advantages. Thus, the header table is arranged in this order of the frequent items.

Construction of FP-Tree

First one scan of the database T is done and L_1, the set of frequent 1-itemsets is computed. For convenience this set is called the set of frequent items. In other words L_1 is referred as a set of items rather than a class of singleton sets. Then the items of L_1 are sorted in the decreasing order of frequency counts. From this stage onwards, the algorithm ignores all the non-frequent items from the transaction and views any transaction as a list of frequent items in the decreasing order of frequency. Without any ambiguity, transaction t can be referred to as such a list. The first element of the list corresponding to any given transaction, is the most frequent item among the items supported by t. For a list t, head_t is denoted as its first element and body_t as remaining part of the list . Thus, t is [head_t | body_t]. The FP-tree construction grows the tree recursively using these concepts.

FP-tree construction Algorithm

```
create a root node root of the FP-Tree and label it as null.
    do for every transaction t
        if t is not empty
        insert (t, root)
        link the new nodes to other nodes with similar labels links originating
        from header list.
    end do
return FP-Tree

insert (t, any_node)
    do while t is not empty
        if any node has a child node with label head_t
        then increment the link count between any_node and head_t by 1
        else create a new child node of any_node with label haed_t with link count 1
        call insert (body_t, head_t)
end do
```

Example: The figure below shows a data set that contains ten transactions and five items. The structures of the FP-tree after reading the first three transactions are also depicted in the diagram. Each node in the tree contains the label of an item along with a counter that shows the number of transactions mapped on to the given path. Initially, the FP-tree contains only the root node represented by the null symbol.

Transaction Data Set

TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b.,c,e}

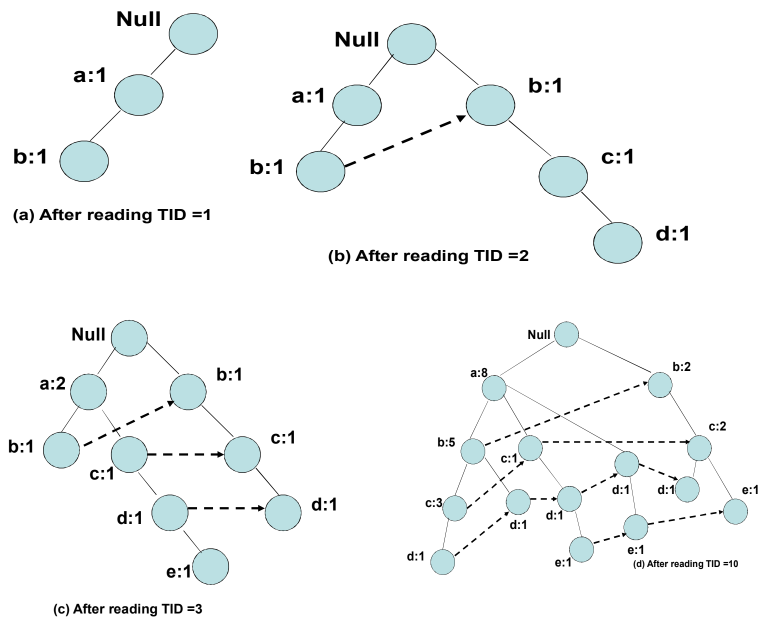


Figure:

Construction of an FP-tree

The FP-tree is subsequently extended in the following way:

1. The data set is scanned once to determine the support count of each item. Infrequent items are discarded, while the frequent items are sorted in decreasing support counts. For the data set shown in the table above, a is the most frequent item, followed by b,c,d and e.
2. The algorithm makes a second pass over the data to construct the FP-tree. After reading the first transaction, {a,b}, the nodes labeled as a and b are created. A path is then formed from null→a→b to encode the transaction. Every node along the path has a frequency count of 1

3. After reading the second transaction, {b,c,d} , a new set of nodes is created for items b,c and d. A path is then formed to represent the transaction by connecting the nodes null→b→c→d. Every node along this path also has a frequency count equal to 1. Although the first two transactions have an item in common, which is b, their paths are disjoint because the transactions do not share a common prefix.

4. The third transaction, {a,c,d,e}, shares a common prefix item (which is a) with the first transaction. As a result the path for the third transaction, null→a→c→d→e, overlaps with the path for the first transaction, null→a→b. Because of their overlapping path, the frequency count for node a is incremented to two, while the frequency counts for the newly created nodes, c, d and e are equal to one.

5. The process continues until every transaction has been mapped onto one of the paths given in the FP-tree. The resulting FP-tree after reading all the transactions is shown in the figure (d).

Stop to Consider

The main disadvantage of the FP tree growth algorithm is that if the database is very large then the algorithm might not fit in the shared memory.

Check Your Progress

1. What is the advantage of FP-tree over apriori algorithm and pincer search algorithm?

2. What is an item prefix tree?

3.4 Generalized Association Rule

Mostly databases consists of numeric values such as age, salary and balance account. So it becomes necessary to find association rules that are applicable both for numeric and categorical attributes. Another feature of association rule generators is an item hierarchy. In the previous chapters we have seen that the examples demonstrated in association rules used generic terms like bread, oregano and mayonnaise. But in a store bread can be sold as a combo pack also. Then new rules might be generated by analysing the fact that differentiates bread sold singly and bread sold in combo pack. In such cases an item hierarchy can be formed to permit grouping of items at different levels. Another aspect that is not taken into account while dealing with association rule is the time. Suppose for example people who buy bed also buy blankets in the time period 2 to 4 months later. Such sequencing rule can be generated where time plays an important part. In market basket problem, support-confidence framework can be used to mine the association rules. In certain cases this framework might not work, for example, negative implications of the type like: when people buy jacket, they don't buy overcoat. But these type of implications might be useful in other situations.

Let us understand this with the help of an example:

Suppose we have market-basket data from a grocery store consisting of n baskets. Let us focus on purchase of cakes and muffins. In the table below, rows c and c contains basket that do and do not contain cakes and similarly the columns m and m corresponds to muffins. The values in the table represent the percentage of baskets.

	c	$\bar{c}$
m	30	10

$\overline{m}$	60	5

Let us apply the support-confidence framework to the potential association rule m⇨c. The support for this rule is 20%, which is fairly high. The confidence is defined to be the conditional probability that a customer buys cake, given that she buys muffins i.e 30/40=0.75 or 75%. which is quiet high. By finding out the confidence, we can conclude that m⇨c is a valid rule.

Let us consider the fact that the apriori probability that a customer buys cake is 80%. In other words it can also be assumed that a customer who buys muffins is less likely to buy cake than a customer about whom we don't have any information. It may still be possible that there may exists a large number of people who may buy cake as well as muffin and there must be certain information to find out this rule. One way to deal such situation is to identify the rules like c⇨m, m ⇨c and c⇨m.

3.5 Association Rules with Item Constraints

In this section we will discuss how to generate candidate sets in the presence of constraints.

Let B be a boolean expression over a set of itemsets. We assume without loss of generality that B is in disjunctive normal form (DNF). That is, B is of the form D_1 V D_2 V……VD_m, where each disjunct D_i is of the form $\alpha_{i1} \wedge \alpha_{i2} \wedge \ldots\ldots\wedge \alpha_{in}$. Each of the α refers either to the presence of an item l_{ij} or to the absence of an item $\neg\, l_{ij.}$

The problem of the mining association rule with item constraint B is to discover all rules that satisfy B and have support and confidence greater than or equal to the user specified minimum support and minimum confidence respectively. We can split the problem into three phases:

Phase 1: This phase comprises of finding all the frequent itemsets that satisfy the Boolean expression B. There are mainly two types of operation: candidate generation and support count. The technique for counting the support remains same but however the apriori candidate generation procedure will no longer generate all the potentially frequent itemsets as candidates when item constraints are present. This subproblem is tackled in the following way:

1. Generate a set of selected items S such that any itemset that satises B will contain at least one selected item.
2. Modify the candidate generation procedure to only count candidates that contain selected items.
3. Discard frequent itemsets that do not satisfy B.

Phase 2: To generate rules fromthese frequent itemsets, we also need to find the support of all subsets of frequent itemsets that do not satisfy B. Recall that to generate a rule ABCD,we need the support of AB to find the confidence of the rule. However, AB may not satisfy B and hence may not have been counted in Phase 1. So we generate all subsets of the frequent itemsets found in Phase 1, and then make an extra pass over the dataset to count the support of those subsets that are not present in the output of Phase 1.

Phase 3: Generate rules from the frequent itemsets found in Phase 1, using the frequent itemsets found in Phases 1 and 2 to compute confidences, as in the Apriori algorithm.

There are various algorithms which are based on both considering and without considering taxonomies over item.

3.6 Summing Up

1. FP-Tree growth algorithm is an efficient algorithm for producing the frequent itemsets without generation of candidate item sets.

2. It is based upon the divide and conquer strategy.

3. A frequent pattern tree (or FP-tree) is a tree structure consisting of an item-prefix tree and a frequent-item-header table.

4. It becomes necessary to find association rules that are applicable both for numeric and categorical attributes.

5. We can also generate association rules with item constraints.

3.7 Answers to Check Your Answer

1. The two advantages of FP-tree growth algorithm over Apriori algorithm is that it is able to handle a large number of candidate sets and it requires only two scan over the database.

2. A frequent pattern tree (or FP-tree) is a tree structure consisting of an item-prefix tree and a frequent-item-header table. An item-prefix treeconsists of a root node labelled null.Each non-root node consists of three fields: item name, support count and node link.

3.8 Possible Questions

1. Define a FP tree.

2. What are the two fields in a frequent item-header table?

3. Mention two features of FP tree.

4. Discuss the method of computing a FP-tree.

5. What is the purpose of generalized association rule?

6. Explain the concept of association rule with item constraint.

3.9 References and Suggested Readings

1. Pujari, A. K. (2001). *Data mining techniques*. Universities press.

2. Tan, P. N., Steinbach, M., & Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

---×---

BLOCK- III

UNIT: 1
INTRODUCTION TO CLUSTERING

Unit Structure:

1.1 INTRODUCTION

In this unit you will get a preliminary idea on cluster analysis. Cluster analysis means dividing a set of data into a number of clusters in such a way that similarity within a cluster is maximized and similarity between any two clusters is minimized. There are different types of clustering methods as well as clustering algorithms to cluster a set of data. Also, the data comes for clustering available in different form. All these above issues have been discussed here.

1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

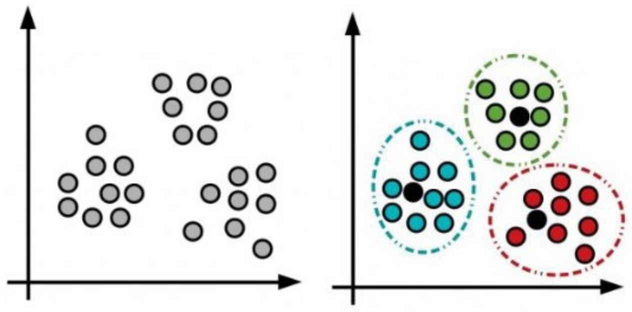
- Understand the concept of data clustering.

- Know about different types of clustering methods.
- Learn about different types of clustering algorithms
- Learn about different types of data available for clustering

1.3 WHAT IS CLUSTERING

Cluster means a group of similar objects or data. Clustering or cluster analysis means dividing a set of data into a number of clusters in such a way that similarity within a cluster is maximized and similarity between any two clusters are minimized.

After clustering a set of data, each cluster contains similar data and data contained in different clusters are dissimilar (Figure 1). Thus, clustering aims at forming groups of homogeneous data from a collection of heterogeneous data, which are unlabeled. So, clustering comes under the unsupervised Learning, which aims at gaining insights from unlabeled data points, i.e. unlike supervised learning no data has a target variable to defined it.



(a)Data before clustering (b) Data after Clustering

Figure 1: Illustration of data clustering

Supervised learning relies on labeled datasets to train algorithms for accurate classification or prediction tasks. These datasets provide clear examples of inputs and their corresponding outputs, allowing the model to learn and improve its performance over time by measuring its accuracy against known outcomes. In contrast, unsupervised learning involves machine learning algorithms that analyze and cluster unlabeled datasets. These algorithms uncover underlying patterns and structures within the data without the need for predefined labels or human guidance. The fundamental difference between supervised and unsupervised learning lies in the presence or absence of labeled data: supervised learning uses labeled input and output data for training, while unsupervised learning operates on unlabeled data.

1.4 TYPES OF CLUSTERING

Clustering broadly divides into two subgroups-

1.4.1 Hard clustering

Hard clustering, also referred to as crisp clustering or traditional clustering, is a method that categorically assigns each data point to exactly one cluster (Figure 2). This assignment relies on similarity or dissimilarity measures, such as Cosine similarity or Euclidean distance, which assess the likeness or disparity between pairs of data points. The primary algorithm associated with hard clustering is K-means clustering, which divides the data into distinct and non-overlapping clusters. This approach is valuable when clearly defined clusters are needed and there is no ambiguity in how data points should be assigned.

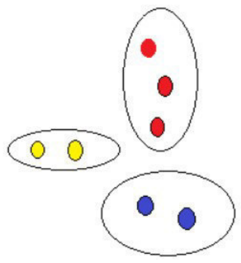


Figure 2: Illustration of hard clustering

1.4.2 Soft Clustering

Soft clustering, also known as fuzzy clustering or probabilistic clustering, offers a more flexible approach to assigning data points to clusters. Unlike hard clustering, where each data point belongs exclusively to one cluster, soft clustering evaluates the probability or likelihood of a data point belonging to each cluster (Figure 3). This means that data can belong to multiple clusters simultaneously, with varying degrees of membership. Soft clustering algorithms like Fuzzy C-means calculate the likelihood of a data point belonging to each cluster based on probabilistic models. Soft clustering accommodates overlapping clusters and captures the uncertainty inherent in data, making it beneficial when data can have multiple interpretations or belong to multiple classes simultaneously.

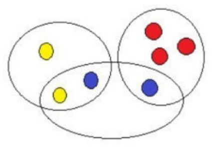


Figure 3: Illustration of soft clustering

1.5 TYPES OF CLUSTERING ALGORITHMS

There are several clustering algorithms available in literature to cluster data. These algorithms can be basically classified into following types-

- Partitional-based Clustering methods
- Hierarchical-based Clustering methods
- Density-based Clustering methods
- Grid-based Clustering methods
- Distribution-based Clustering

1.5.1 Partitional-based Clustering methods

Partitional clustering (Figure 4) is simply a division of the set of data into non-overlapping clusters such that each data is in exactly onecluster. The data are clustered into pre-defined k number of clusters based on the closeness of the data. Each cluster is represented using a cluster representative called prototype of the cluster. In partitional clustering methods, each data belonging to a cluster is more similar to the prototype that defines the cluster than to the prototype of any other cluster. Here, the similarity is measured using any dissimilarity or similarity measure based on the data and based on the types of the data, the cluster prototype may be the centroid of the cluster, medoid of the cluster etc. The popular clustering algorithms K-Means and K-Medoids are examples of such types of clustering algorithms. This method is suitable when clusters present in data are of regular shape. The primary drawback for these algorithms is that the number of clusters, k, must be predefined, which requires prior knowledge of the data. Due to its preset number of cluster requirements and extreme sensitivity to the initial positioning of centroids, the outcomes can vary. Furthermore, the tendency of centroid-based approaches to produce spherical or convex clusters restricts their capacity to handle complicated or arbitrary or irregularly shaped clusters.

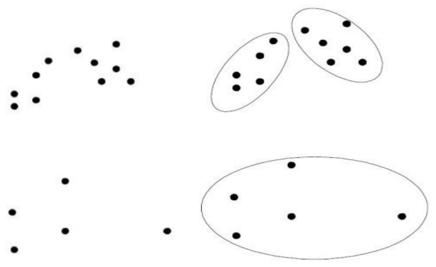
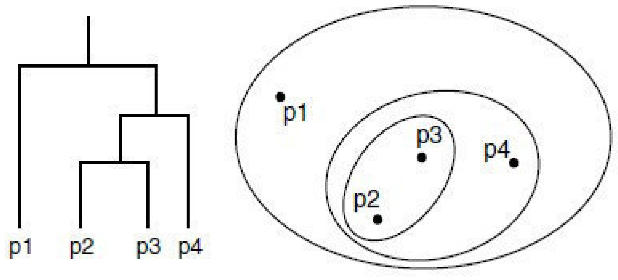

(a) Original Data (b) Partitional clustering with k=3

Figure 4: Illustration of partitional clustering

1.5.2 Hierarchical-based clustering methods

In hierarchical clustering methods (Figure 5), the clustered data are obtained in a hierarchical form, which is often, displayed graphically using a tree-like diagram called a dendrogram. Thus, the dendrogram shows the hierarchical relationships among the clusters. Initially each data is treated as a separate cluster at the base of the tree and iteratively the closest pairings of clusters are then merged into larger clusters. When every object is in one cluster at the top of the tree, the merging process has finished. To obtain a given number of clusters, the dendrogram can be cut at a particular height, using some stopping criterion in the merging process.



(a) Dendrogram (b) Nested cluster diagram

Figure 5: Illustration of hierarchical clustering

There aretwo basic approaches for generating a hierarchical clustering: Agglomerative and Divisive

A. Agglomerative: It follows a bottom-up approach. It starts with the points as individual clusters and, at each step, merges the closest pair of clusters. This requires defining a notion of cluster proximity. Many agglomerative hierarchical clustering techniques are variations on a single approach given in Algorithm 1. The proximity can be measured in terms of distance measure, similarity measure etc.

Algorithm 1: Basic agglomerative hierarchical clustering

```
Step 1: Compute the proximity matrix, if necessary.
Step 2:  Repeat
Step 3: Merge the closest two clusters.
Step 4: Update the proximity matrix to reflect the proximity
            between the new cluster and the original clusters.
Step 5: Until only one cluster remains.
```

The proximity can be measured as described below-

i. Single Linkage (SL)

The distance between two groups G1 and G2 is defined as the minimum distance between any element i of G1 and any element j of G2 as shown below-

$$D_{SL}(G1, G2) = \min_{i \epsilon G1, j \epsilon G2} d(i, j)$$

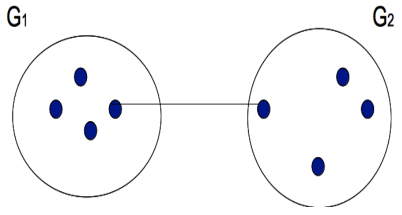


Figure 6: Illustration of Single linkage

Clusters formed using the Single Linkage method often exhibit long and narrow shapes and tend to lack internal homogeneity. Due to these characteristics, this method is not suitable for identifying overlapping clusters but performs well when clusters are distinctly separated from each other.

ii. Complete Linkage (CL)

The distance between two groups G1 and G2 is defined as the maximum of the distances between any element i of G1 and any element j of G2 as shown below-

$$D_{CL}(G1, G2) = \max_{i\epsilon G1, j\epsilon G2} d(i, j)$$

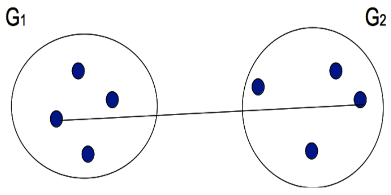


Figure 7: Illustration of Complete linkage

Clusters yielded by this procedure tend to be spherical and compact; nothing can be said about their separation.

iii. Average Linkage (AL)

The distance between two groups G1 and G2 is defined as the arithmetic means of the distances between any elements i of G1and any element j of G2:

$$D_{AL}(G1, G2) = mean\ d(i, j)$$

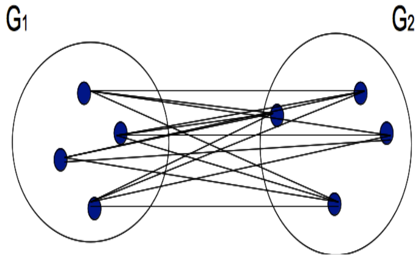


Figure 8: Illustration of Average linkage

iv. Centroid Linkage

The distance between two groups G1 and G2 is defined as the Euclidean distance between the two cluster centers (the so-called centroids or prototypes), i.e., the centroid or prototype of a cluster is the mean vector of the observed variables computed using the units assigned to the cluster.

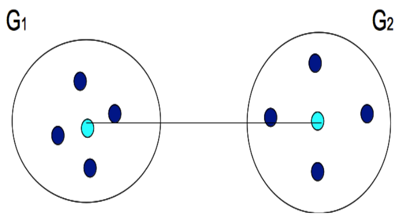


Figure 9: Illustration of Centroid linkage

One issue that can arise with this approach is that the dissimilarity of a union is lower than the dissimilarity measured at the previous merge, which can lead to less interpretable results.

B. Divisive: It follows a top-down approach. It starts with one, all-inclusive cluster and, at each step, split a cluster until only singleton clusters of individual points remain. In this case, some criteria are needed to decide which cluster to split at each step and how to do the splitting.

1.5.3 Density-based clustering methods

Density-based clustering is a model-based method that identifies groups based on the density of data points. Unlike centroid-based clustering, which requires a predefined number of clusters, density-based clustering determines the number of clusters automatically. Moreover, this approach is well-suited for datasets with arbitrary shape, irregular shapes or overlapping clusters. By focusing on local density, it effectively handles both dense and sparse regions in the data and can detect clusters of diverse shapes and sizes. Therefore, density-based clustering addresses the limitations of centroid-based techniques by autonomously determining cluster sizes, being robust to initialization issues, and successfully identifying clusters with varying shapes and sizes. DBSCAN is one of the most widely used algorithms for density-based clustering.

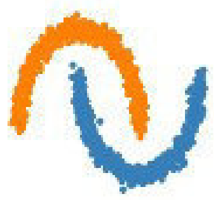


Figure 10: An example of arbitrary shaped data

1.5.4 Grid-Based Methods

Grid-based clustering methods partition the object space into a finite number of cells organized in a grid structure. All clustering operations are then executed within this grid structurei.e. on quantized space. The primary benefit of this approach lies in its rapid processing time, which generally remains consistent regardless of the number of data objects. Instead, the computational load depends solely on the number of cells in each dimension of the grid. STING a typical example a common grid-based clustering method used in practice.

1.5.5 Distribution based Clustering

Distribution-based clustering methods identify clusters based on the statistical distribution of data points. These methods assume that the data points are generated from a mixture of probability distributions, typically Gaussian distributions or other parametric models. The goal is to model the underlying distribution of the data and use it to identify clusters. Examples of distribution-based clustering algorithms include Gaussian Mixture Models (GMMs), which are widely used for their ability to model complex data distributions and handle overlapping clusters effectively.

1.6 DIFFERENT TYPES OF DATA USED FOR CLUSTERING

Clustering algorithms can be applied to various types of data, each with its own characteristics. Here are different types of data commonly used for clustering:

i. **Numerical Data**: This is the most common type of data used in clustering. Numerical data consists of quantitative values that can be measured on a continuous scale. Examples include measurements such as height, weight, temperature, and financial metrics. Clustering algorithms like K-means, hierarchical clustering is often applied to numerical data.

ii. **Categorical Data**: Categorical data consists of qualitative variables that represent categories or groups. Examples include types of products, customer segments, or yes/no responses.

Clustering categorical data requires special techniques, such as K-Mode clustering or methods like one hot encoding that convert categorical variables into numerical representations.

iii. **Mixed Data**: Many real-world datasets contain a combination of numerical, categorical or other variables. These are referred to as mixed data. Clustering mixed data requires algorithms that can handle both types of variables simultaneously, such as K-prototype clustering or methods that transform categorical data into numerical form (e.g., one-hot encoding) for use with traditional numerical clustering algorithms.

iv. **Text Data**: Text data includes documents, emails, reviews, etc., represented as strings of words or tokens. Text clustering involves techniques like term frequency-inverse document frequency (TF-IDF) to convert text into numerical vectors, followed by clustering algorithms such as K-means or hierarchical clustering.

v. **Image Data**: Image clustering involves grouping similar images together based on their visual features. Techniques like feature extraction (e.g., using convolutional neural networks) are used to transform images into numerical representations, which are then clustered using algorithms suitable for numerical data.

vi. **Spatial Data**: Spatial data involves geographical or spatial coordinates, such as locations on a map or coordinates in a 3D space. Spatial clustering algorithms consider the distance or spatial relationships between data points. Algorithm like DBSCAN is used for spatial data.

vii. **Temporal Data**: Temporal data includes time-series data, where each data point is associated with a timestamp or a sequence of time-stamped events. Clustering temporal data involves techniques that consider the temporal ordering and dependencies between data points over time, such as dynamic time warping (DTW) for similarity measurement or time series clustering algorithms.

Each type of data presents unique challenges and requires appropriate pre-processing and clustering techniques to derive meaningful insights and patterns. The choice of clustering algorithm depends on the nature of the data and the specific objectives of the analysis.

Check Your Progress-1

1. What is cluster?

2. What is cluster analysis?

3. What is type of clustering?

4. What is partition-based clustering?

5. What is hierarchical-based clustering?

6. Mentioned different ways that can be used to measure proximity in agglomerative clustering.

7. What is Density-based clustering?

8. What is grid-based clustering?

9. What is distribution-based clustering?

10. Mentioned different types of data used for clustering.

State TRUE or FALSE:

11.Clustering means grouping of similar objects or data.

12. K-Means is a partition-based clustering.

13. DBSCAN is suitable when clusters are of regular shapes.

14. Agglomerative clustering is a top to bottom down approach.

15. Categorical data can be converted into numeric form using one hot encoding technique.

1.7 SUMMING UP

- **Cluster** means a group of similar objects or data.

- **Clustering or cluster analysis** means dividing a set of data into a number of clusters in such a way that similarity within a cluster is maximized and similarity between any two clusters is minimized.

- The fundamental difference between **supervised and unsupervised learning** lies in the presence or absence of labeled data: supervised learning uses labeled input and output data for training, while unsupervised learning operates on unlabeled data.

- **In hard clustering** each data point is assigned to exactly one cluster.

- In **soft clustering** each data can belong to multiple clusters simultaneously, with varying degrees of membership.

- **Partitional clustering** is simply a division of the set of data into non-overlapping K numbers of clusters such that each data is in exactly one cluster.

- **K-Means** and **K-Medoids** are examples of partitional clustering methods.

- **Partitional clustering method** is suitable when clusters present in data are of regular shape

- In **hierarchical clustering methods** the clustered data are obtained in a hierarchical form, which is often displayed graphically using a tree-like diagram called a dendrogram.

- There are two basic approaches for generating a hierarchical clustering: **Agglomerative and Divisive**

- **The agglomerative** method follows a bottom-up approach. It starts with the points as individual clusters and, at each step, merges the closest pair of clusters.

- In agglomerative clustering the **proximity can be measured** using the methods- Single Linkage, Complete Linkage, Average Linkage, Centroid Linkage.

- **The divisive** method follows a top-down approach. It starts with one, all-inclusive cluster and, at each step, split a cluster until only singleton clusters of individual points remain.

- **Density-based clustering** is a model-based method that identifies groups based on the density of data points.

- **Unlike centroid-based clustering**, which requires a predefined number of clusters, density-based clustering determines the number of clusters automatically.

- **Density-based** approach is well-suited for datasets with arbitrary shape, irregular shapes or overlapping clusters.

- **Grid-based clustering** methods partition the object space into a finite number of cells organized in a grid structure. All clustering operations are then executed within this grid structurei.e. on quantized space.

- **Distribution-based clustering** methods identify clusters based on the statistical distribution of data points.

- **Numerical data** consists of quantitative values that can be measured on a continuous scale.

- **Categorical data** consists of qualitative variables that represent categories or groups.

- **Clustering categorical data requires** special techniques, such as K-Mode clustering or methods like one hot encoding that convert categorical variables into numerical representations.

- **Mixed data** contains a combination of numerical, categorical or other variables.
- **Text data** includes documents, emails, reviews, etc., represented as strings of words or tokens.

- **Image clustering** involves grouping similar images together based on their visual features.

- **Spatial data** involves geographical or spatial coordinates, such as locations on a map or coordinates in a 3D space.

- **Temporal data** includes time-series data, where each data point is associated with a timestamp or a sequence of time-stamped events.

1.8ANSWERS TO CHECK YOUR PROGRESS

State TRUE or FALSE:

11. True.
12. True.
13. False.
14. False.
15. True

1.9 POSSIBLE QUESTIONS

Short answer type questions:

1. What is cluster analysis? Explain different types of clustering methods.
2. Describe partitional clustering method.
3. Describe different types of ways to measure proximity of data in agglomerative clustering.
4. Describe density-based clustering.
5. Write the difference between partitional and density-based clustering.
6. Describe grid-based clustering.
7. Describe distribution-based clustering.

Long answer type questions:

1. Write about different types of clustering algorithm.
2. Write about different types of hierarchical clustering methods.
3. Write about different types of data available for data clustering.

1.9 REFERENCES AND SUGGESTED READINGS

1. “Data Mining Techniques” by Arun Kumar Pujari.

2. “Introduction to Data Mining” by Pang-Ning Tan, Michael Steinbach, Vipin Kumar

---×---

UNIT: 2
PARTITIONING CLUSTERING ALGORITHMS

Unit Structure:

2.1 Introduction

As we know now the concept of clustering, let us discuss in this chapter one of the types of clustering algorithm i.e., Partitioning Clustering Algorithm. We will study the underlying principles of the partitional clustering algorithms. The two main categories of partitional algorithms are K-means and K-medoid algorithms. There are different types of k-medoid algorithms like PAM, CLARA and CLARANS which will be discussed in details in this chapter.

2.2 Unit Objectives

In this unit you will be able to learn

- The underlying principle of Partitioning algorithms
- Categories of Partitioning Algorithms
- Difference between k-means and k-medoid algorithms
- Types of different k-medoid algorithms

2.3 Partitioning Clustering Algorithms

The concept of Partitioning Clustering algorithm is to construct partitions of a database of N objects into a set of clusters. This construction is accomplished to determine the optimal partition with respect to a certain objective function. The number of ways in which a database can be partitioned is kN/k! ways which means partitioning N data points into k subsets. To find the global optimal partition, is practically infeasible when N and k are very large. In such cases, Iterative Optimization paradigm is adopted. This paradigm starts with an initial partition and uses an iterative control strategy. It analyses that whether swapping of the data points improves the quality of clustering. This is continued until swapping does not yield any improvements. Once such happens, it finds a locally optimal partition. Thus in this type of clustering the initial selected partitions are very significant. There are two types of partitioning clustering algorithms: K-means and K-medoid.

2.3.1 K-means Algorithms

K-means algorithm represents each cluster using the center of gravity of the cluster. It is an unsupervised learning algorithm as the data points does not have any class labels. The algorithm divides the data points into clusters based on the similarities of the data points. The data points present in a cluster are similar to one another but they differ from the other clusters. Thus it tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. Here K is the number of clusters which is decided by the user based on the type of application used. The first step in K-means algorithm is to assign two points as centroid. Based on the distance of the data points from the centroid, the clusters are being formed. The Basic K-means Algorithm for finding K clusters is given below:

1. Select K points as the initial centroids.

2. Assign all points to the closest centroid.

3. Recompute the centroid of each cluster. The flowchart of the K-means algorithm is depicted Fig 1.

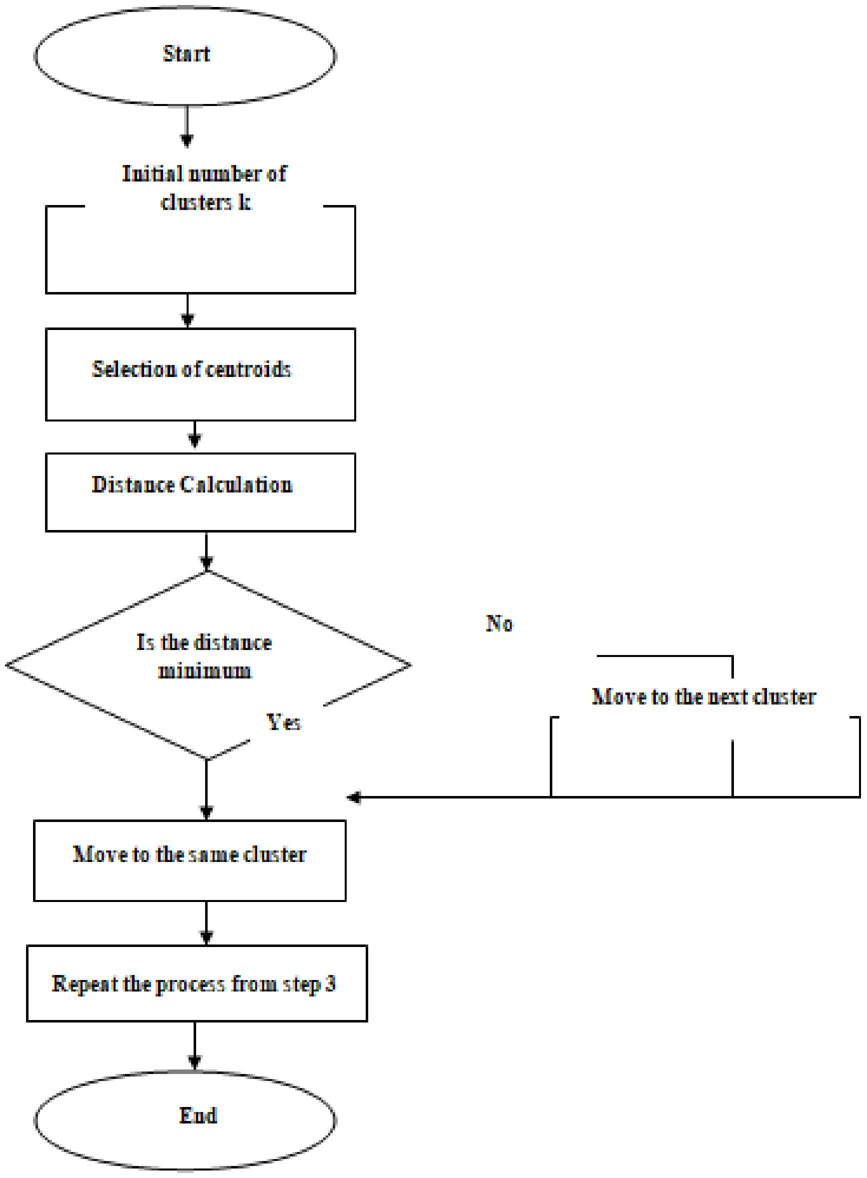


Fig 1: Flowchart of K-means algorithm

Advantages of K-means algorithm

1. This algorithm is a popular choice for clustering task as it is simple and easy to implement.
2. K-means is a very efficient algorithm as it can handle large datasets with high dimensionality without taking much time.
3. K-means can handle large datasets with a large number of data points and can be easily scaled to handle even larger datasets.
4. K-means can be easily adapted to different applications and can be used with different distance metrics and initialization methods.

Disadvantages of K-means algorithm

1. K-means is sensitive to the initial selection of centroids and can converge to a suboptimal solution.
2. The number of clusters k needs to be specified before running the algorithm, which can be challenging in some applications.
3. K-means is sensitive to outliers, which can have a significant impact on the resulting clusters.

Applications of K-means algorithm

Academic Performance: Based on the marks obtained by the students, their results are categorized into grades like A, B, or C.

Diagnostic systems: The patients having same symptoms of a particular disease can be separated from those not having it. Thus helps in making smarter diagnostic systems

Search engines: The search operations can be grouped based on the similarity which helps the search engine further in retrieving the results. Thus clustering forms, the backbone of a search engine.

Wireless sensor networks: The clustering algorithm plays the role of finding the cluster heads, which collect all the data in its respective cluster.

STOP TO CONSIDER

K-means algorithm gives higher priority to bigger clusters.

2.3.2 K-medoid Algorithms

In K-medoid algorithm, each cluster is represented by one of the objects of the cluster located near the center.

2.3.2.1 PAM (Partition Around Medoids)

PAM(**P**artition **A**round **M**edoids) uses a K-medoid method to identify the clusters.K objects are arbitrarily chosen as medoids from the given data.Medoid is the most centrally located object in a cluster. Each of this k objects are representatives of k classes. The main aim of this algorithm is to minimize the sum of the dissimilarities of the objects to their closest representative objects. The basic steps in PAM is given below:

1. Select K initial points. These points are the candidate medoid and are intended to be the most central point of their clusters.

2. Considering the effect of replacing one of the selected objects with one of the non-selected objects.

The flowchart of the PAM algorithm is depicted in Fig 2:

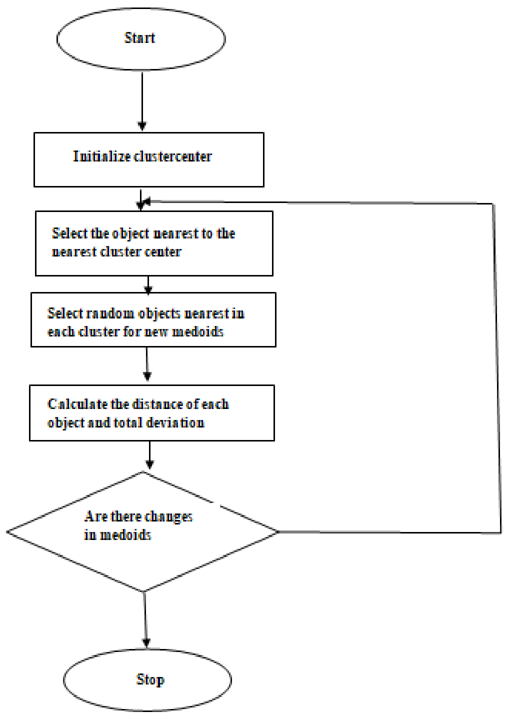

Start
Initialize clustercenter
Select the object nearest to the nearest cluster center
Select random objects nearest in each cluster for new medoids
Calculate the distance of each object and total deviation
Are there changes in medoids
Stop

Fig 2: Flowchart of K-medoid algorithm

The algorithm of PAM is given below

PAM Algorithm

```
Input: Database of objects D.
select arbitrarily k representative objects, $K_{med}$
mark these objects as "selected" and mark the remaining as "non-selected"
do for all selected object $O_i$
do for all non-selected objects $O_h$
        compute $C_{ih}$
end do
end do
select $i_{min}$, $h_{min}$ such that $C_{imin,hmin} = Min_{i,h} C_{ih}$
if $C_{imin,hmin} < 0$
    then swap: mark $O_i$ as non-selected and $O_h$ as selected.
repeat
find clusters $C_1, C_2, C_3 \ldots\ldots, C_k$
```

The algorithm starts by selecting k-medoids and it iteratively improves upon this selection. At every step a swap takes place between a selected object O_i and a non-selected object $O_h$is made. These swaps continue as long as such swap improves the quality of clustering. To find the effect of such a swap, a cost$C_{ih}$is computed which is related to the quality of partitioning the non-selected objects to k clusters represented by the medoids. The algorithm has two important phases: partitioning of the database for a given set of medoids and the iterative selection of medoids.

Partitioning:

If O_j is a non-selected object and O_i is a medoid, we then say that $O_j$belongs to the cluster represented by $O_i$if $d(O_i, O_j) = \text{Min}_{O_e} d(O_j, O_e)$, where the minimum is taken overall medoids $O_e$and $d(O_a, O_b)$

determines the distance or dissimilarity, between objects O_a and O_b .Before initializing the algorithm, the dissimilarity matrix to be used is decided. The average dissimilarity between an object and the medoid of the cluster to which the object belongs determines the quality of the clustering.

Iterative Selection of Medoids

Let there be k-medoids$O_1, O_2, \ldots\ldots O_k$ at any stage and C_1, $C_2, \ldots\ldots\ldots C_k$ be the respective clusters. For a non-selected object O_j , $j \neq 1,2,3\ldots k$

if $O_j \in C_m$ then $\mathrm{Min}_{(1 \leq i \leq k)}\, d\,(O_j, O_i) = d\,(O_j , O_m)$

Let us analyze the effect of such a swapping. Suppose an unselected object O_h becomes a medoid which replaces an existing medoid O_i . Now the new set of medoids is represented by Kmed = $\{O_1 , O_2 , \ldots\ldots\ldots, O_{i-h} , O_h , O_{i+1} , \ldots..O_k\}$ where O_h replaces $O_i$as one of the medoids from Kmed = $\{ O_1, O_2 ,\ldots\ldots\ldots O_k\}$. Now let's make a comparison on the quality of clustering. Due to the change in the medoids mentioned above, there will be three types of changes that might occur in the actual clustering. Let C_h denote the new clusters with O_h as the representative.

1. A non-selected object O_j such that $O_j \in C_i$before swapping and $O_j \in C_i$after swapping. In this type of cases the following cases hold:

Min $d(O_j , O_e) = d\,(O_j, O_i)$ where minimum taken over O_e in Kmed and

$\mathrm{Min}_{e \neq i}\, d(O_j , O_e) = d\,(O_j, O_h)$ where minimum taken for all O_e in Kmed

Let us define a cost for this change as

$C_{jih} = d(O_j, O_h) - d(O_j, O_i)$

2. A non-selected object $O_j \in C_i$before swapping and $O_j \in C_g$, $g \neq h$, after swapping. This case arises when

Min $d(O_j , O_e) = d\,(O_j, O_i)$ where minimum taken over O_e in Kmed and

Min $d(O_j , O_e) = d\,(O_j, O_g)$ where minimum taken over O_e in Kmed

Again a cost C_{jih} must be defined

$C_{jih} = d(O_j, O_g) - d(O_j, O_i)$

3. A non-selected object $O_j \in C_i$before swapping and $O_j \in C_h$ after swapping. This case arises when

Min $d(O_j , O_e) = d (O_j, O_g)$ where minimum taken over O_e in Kmed and

Min $d(O_j , O_e) = d (O_j, O_h)$ where minimum taken over O_e in Kmed

A cost for this case is like

$C_{jih} = d(O_j, O_h) - d(O_j, O_g)$

From the last three cases, only one case might occur for any non-selected object O_j .

The sum of all the cost of all non-selected objects is the total cost of swapping two medoids.

Let us now see how PAM algorithm works when a swapping takes place.

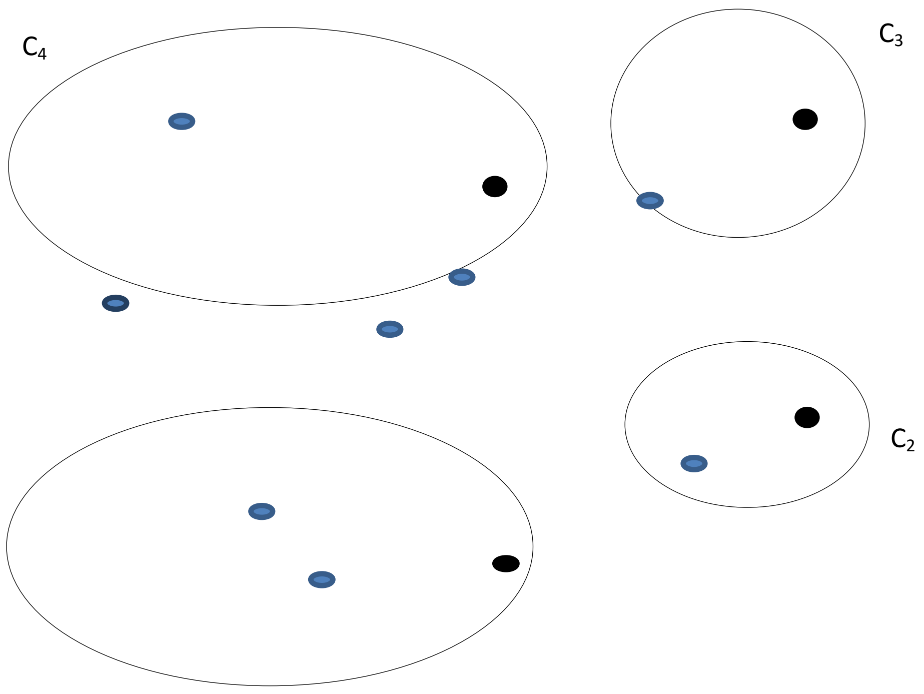


Fig: Partition Before Swapping

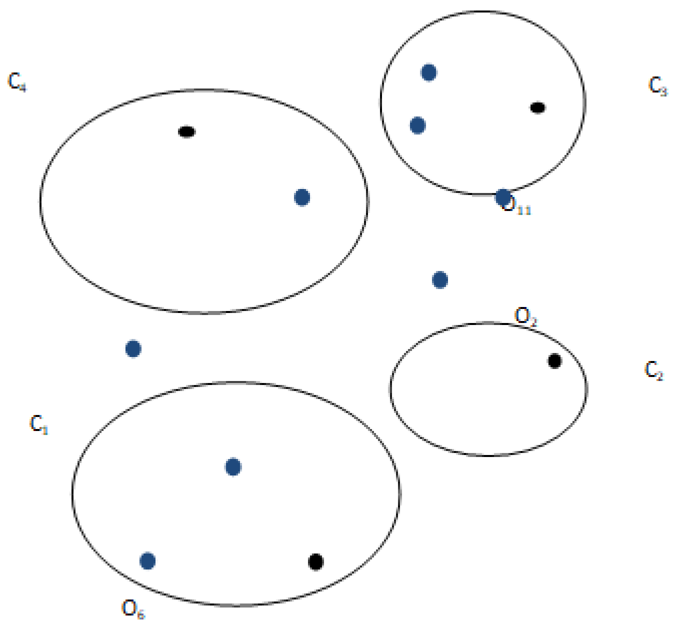


Fig: Clustering after Swapping O_5 by O_4

In Fig we can see that the initial selected medoids are O_1 , O_2 , O_3 and O_5. Based on the closest distance with the medoid the remaining objects are newly classified as shown in Fig . Let's assume that O_4 is the new medoid in place of O_5. The distance metric here is denoted by d_{ij} (O_i ,O_j) and is considered to be symmetric. The probable three cases are mentioned below:

Case 1: The object O_8 was in cluster C_5 before swapping and in C_4 after swapping.

Therefore, the cost $C_{854} = d_{58} – d_{48}$

Case 2: The object O_9 was in C_5 before swapping and in C_1 after swapping.

Hence the cost for O_9 is $C_{954} = d_{19} – d_{59}$

Case 3: The object O_7 was in C_1 before swapping and in C_4 after swapping.

So the cost for O_7 is $C_{754} = d_{47} – d_{17}$

The total cost of swapping O_i and O_h is defined as

$C_{ih} = \Sigma_j C_{jih}$ where the sum is taken for all objects j.

If the result of C_{ih} is negative, then the medoid selected is changed by making O_h as medoid in place of O_i . The process of computing C_{ih} is continued until we cannot find a negative C_{ih}. If more than one pair (i,h) of negative cost is found then the pair corresponding to the most negative C_{ih} is chosen.

Advantages of PAM algorithm

1. Simple and easy to implement.

2. Fast algorithm as it converges in a fixed number of steps.

3. It is less sensitive to noise and outliers compared to other partitioning algorithms.

Disadvantages of PAM algorithm

1. This type of algorithm is not suitable for clustering arbitrarily group of data points.

2. Different runs of the algorithm give different results as it depends on the initial medoids which are chosen randomly.

STOP TO CONSIDER

PAM is very difficult to implement and is slower than k-means algorithm. For these reason researchers developed algorithms like CLARA and CLARANS, which increases the speed of clustering at the cost of optimal assignment of clusters

2.3.2.2 CLARA (Clustering Large Applications)

CLARAis an extension of PAM algorithm. The computational cost of determining the medoids in PAM through iterative optimization is very expensive. CLARA minimizes the computational effort by drawing a sample of the dataset and applying PAM on them to determine the medoids rather then finding representative objects for the entire dataset. By using the partitioning principle, the remaining objects are then classified. Thus the medoids of the sample would approximately represent the medoids of the entire dataset. The CLARA algorithm is given in Fig

CLARA Algorithm

```
Input: Database of D objects
repeat for m times
        draw a sample $S \subseteq D$ randomly from D
        call PAM (S,k) to get k medoids
        classify the entire dataset D to $C_1, C_2, \ldots\ldots C_k$
calculate the quality of clustering as the average dissimilarity.
end
```

Advantages of CLARA Algorithm

1. CLARA can deal with large datasets compared to PAM.

Disadvantages of CLARA Algorithm

1. CLARA's performance depends upon the size of the dataset.

2. A sample selected biasedly to represent the whole dataset may result in poor clustering.

2.3.2.3 CLARANS (Clustering Large Applications based on Random Search)

CLARANS is usually used for spatial data. It is similar to PAM and CLARA but for determining medoids it applies a randomized Iterative-Optimization method. CLARANS overcomes the disadvantage of both PAM and CLARA algorithm. It takes care of both the computational cost and also the effect of data sampling on clusters formation. The steps of the CLARANS algorithm are as follows:

1. Select 'k' random data points and label them as medoids for the time being.

2. Select a random point say 'a' from the points picked in step (1), and another point say 'b' which is not included in those points.

3. We would already have the sum of distances of point ‘a’ from all other points since that computation is required for selecting the points in step (1). Perform similar computation for point ‘b’.

4.If the sum of distances from all other points for point ‘b’ turns out to be less than that for point ‘a’, replace ‘a’ by ‘b’.

5. The algorithm performs such a randomized search of medoids ‘x’ times where ‘x’ denotes the number of local minima computed, i.e. number of iterations to be performed, which we specify as a parameter. The set of medoids obtained after such ‘x’ number of steps is termed as ‘local optimum’.
6. A counter is incremented every time a replacement of points is made. The process of examining the points for possible replacement is repeated till the counter does not exceed the maximum number of neighbors to be examined (specified as a parameter).

7. The set of medoids obtained when the algorithm stops is the best local optimum choice of medoids.

The algorithm of CLARANS algorithm is given in Fig

CLARANS Algorithm

```
Input(D,k, maxneighbour and numlocal)

select arbitrarily k representative objects

mark these objects as “selected” and all other objects as non-
selected. Call it current.

set e=1

do while ($e \leq$ numlocal)

      set j =1

do while ($m \leq$ maxneighbour)

            select randomly a pair (j,h) such that $O_j$ is a selected object
and $O_h$ is a non-selected

            object.

           compute the cost $C_{jh}$
```

```
if $C_{jh}$ is negative
            "update current"
              Mark $O_j$ non-selected, $O_h$ selected and m=1
else
            increment m ← m+1
end do
        compare the cost of clustering with "mincost"
        if current_cost < mincost
            mincost ←current_cost
            best_node ← current
increment  e ← e+1
end do
return "best node"
```

Advantages of CLARANS algorithm

1. It handles outliers efficiently

2. More effective compared to PAM and CLARA

Disadvantages of CLARANS algorithm

1. It assumes that all objects fit in main memory and the result is sensitive to the input order.

2. It may not find a real local minimum due to the trimming of it's searching.

CHECK YOUR PROGRESS

1. Partitioning Clustering algorithm is to construct partitions of a database of N objects into a set of ____________

2. _____________ paradigm is adopted in Partitioning algorithms.

3. ______________ algorithm represents each cluster using the center of gravity of the cluster.

4. K-means is sensitive to _______, which can have a significant impact on the resulting clusters.

5. In PAM, K objects are arbitrarily chosen as _______ from the given data.

6. _________ algorithm is not suitable for clustering arbitrarily group of data points.

7. The performance of ________ algorithm depends upon the size of the dataset.

8.The _________ algorithm takes care of both the computational cost and also the effect of data sampling on clusters formation.

2.4 SUMMING UP

1. Partitional clustering attempts to directly decompose the dataset into a set of disjoint clusters.

2. The quality of the cluster is improved by swapping elements.

3. There are mainly two types of partitional algorithms: k-means and k-medoid.

4. The algorithms that fall under k-medoid algorithm are: PAM, CLARA and CLARANS

5. PAM uses a K-medoid method to identify the clusters.

6. Medoid is the most centrally located object in a cluster.

7. CLARA is an extension of PAM algorithm.

8. CLARA draws a sample of the dataset and applies PAM on them to determine the medoids.

9. CLARANS combines the approach of both PAM and CLARA.

10. CLARANS applies a randomized Iterative Optimization method for determining medoids.

2.4 Answers to Check Your Progress

1. clusters

2. Iterative Optimization

3. k-means

4. outliers

5.medoids

6. PAM

7. CLARA

8. CLARANS

2.5 Possible Questions

1. What is the logic behind the Partitioning algorithm?

2. What are the two types of partitioning clustering algorithm?

3. List two advantage and disadvantage of k-means algorithm?

4. Mention two applications of k-means algorithm?

5. State the basic steps of k-means algorithm

6. Draw the flowchart of k-means algorithm?

7. What is a medoid?

8. What is the aim of PAM algorithm?

9. How the medoids are chosen in PAM?

10. State the working of the PAM algorithm with the help of an example.

11. How is CLARA different from PAM algorithm?

12. List two demerits of CLARA algorithm.

13. Describe the CLARANS algorithm.

14. List two merits of CLARANS algorithm

2.6 References and Suggested Readings

[1]Pujari, A. K. (2001). *Data mining techniques*. Universities press

[2] Batra, A. (2011, November). Analysis and approach: K-means and K-medoids data mining algorithms. In *ICACCT, 5th IEEE International Conference on Advanced Computing & Communication Technologies* (pp. 274-279).

[3] Kandali, K., Bennis, L., & Bennis, H. (2021). A new hybrid routing protocol using a modified K-means clustering algorithm and continuous hopfield network for VANET. *IEEE Access*, *9*, 47169-47183.

[4]https://www.researchgate.net/publication/346005407/figure/fig3/AS:960161757814786@1605931810981/K-medoids-algorithm-flowchart.png

[5]https://analyticsindiamag.com/developers-corner/comprehensive-guide-to-clarans-clustering-algorithm/

[6]https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

[7]https://www.simplilearn.com/tutorials/machine-learning-tutorial/k-means-clustering-algorithm

[8]https://www.kdnuggets.com/2020/12/algorithms-explained-k-means-k-medoids-clustering.html

---x---

UNIT: 3
HIERARCHICAL ALGORITHMS

Unit Structure:

3.1 Introduction

In the previous chapter, we have learned about partitioning algorithm, it's concept and the various algorithms that belong to it's category. In this unit we will learn about the hierarchical algorithms. As we know that there are some challenges in K-means clustering like it predetermines a number of clusters, and it always tries to create the clusters of the same size. In order to overcome these two challenges, hierarchical clustering algorithm were developed because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

3.2 Unit Objectives

In this unit you will be able to learn

- The underlying principle of Hierarchical algorithms
- Type of Hierarchical algorithm
- The logic behind BIRCH and CURE algorithms

3.3 Hierarchical Algorithms

Hierarchical algorithm creates a hierarchical decomposition of the database by grouping the data points that are similar. The process of this algorithm starts by finding the data points that are close to one another and are placed in a cluster. The data points that are close to each other are considered to be similar while the other data points are considered to be dissimilar and are placed in other clusters. In this way clustering is continued until all the data points are grouped into clusters based on their similarity. Thus at the end we get a hierarchical tree of related groups known as dendrogram. A dendrogram is a tree-like structure that explains the relationship between all the data points in the system. An example of a dendrogram is depicted in Fig 1.

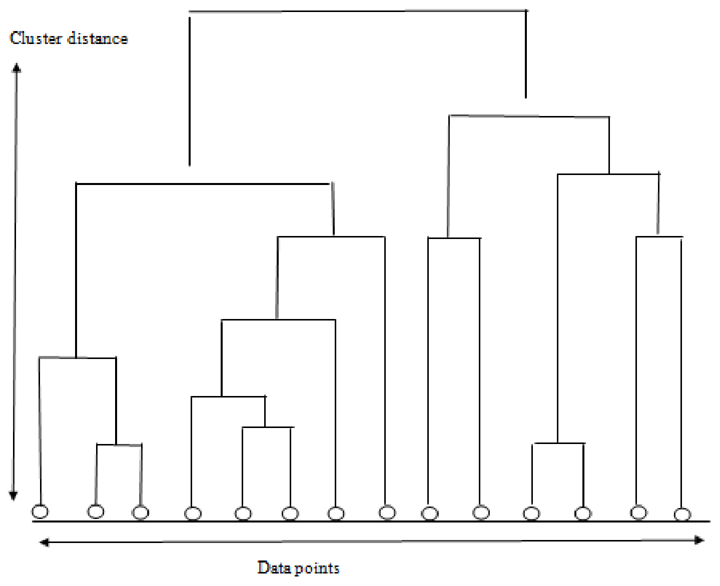


Fig 1 : Dendrogram with data points on the x-axis and cluster distance on the y-axis

This dendrogram can be represented in two ways:

1. **Bottom-up (Agglomerative) approach:** It is a bottom-up approach. It starts with each object being placed in a unique cluster and then merging iteratively the clusters that are similar to one another, thus forming a large cluster. It continues until a terminating

condition specified by the user is achieved or all clusters merge into a single cluster. An example of Agglomerative clustering is depicted in Figure 2.

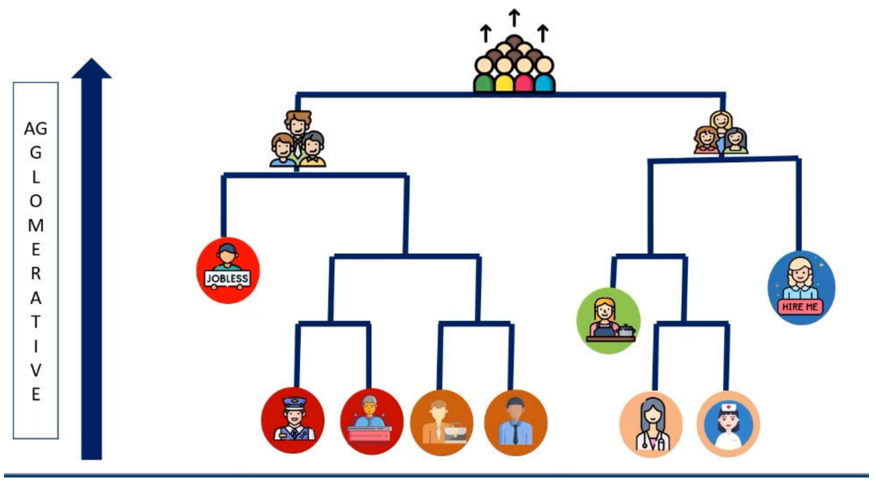


Fig 2: Agglomerative Clustering Example

2. **Top-down (Divisive) approach**: It is a top-down approach. It starts with placing all the objects in a large cluster and then gradually splitting it into smaller and smaller cluster. It continues until a user specified condition is met or all objects are placed in a single cluster. An example of Divisive clustering is depicted in Figure 3.

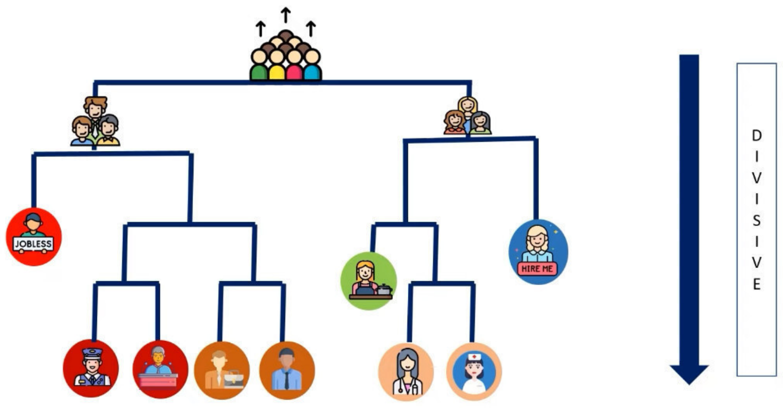


Fig 3: Divisive Clustering Example

Advantages of Hierarchical Algorithms

1. The number of clusters need not be specified.
2. It is easy to understand and interpret.
3. It can handle large datasets without being computationally expensive and time consuming.
4. As hierarchical algorithms produce a virtual tree structure providing greater insight and more information that is easily understood.
5. It is flexible enough to handle any type of data.

Disadvantages of Hierarchical Algorithms

1. It is sensitive to noise and outliers.
2. It cannot handle categorical values.
3. It cannot guarantee optimal results.
4. It is difficult to handle different sized clusters and convex shapes.
5. There is a probability of producing overlapping clusters in case of agglomerative and divisive clustering.

Applications of Hierarchical Algorithms

1. Customer segmentation: Hierarchical Clustering can be used to group customers in different clusters depending upon the buying patterns, demographic information. This can be of great help in studying the customer behavior for target making campaigns.
2. Image segmentation: It can also be used to segment images into different regions based on their dissimilar characteristics.
3. Text analysis: Hierarchical clustering can also be used in grouping documents based on the text content and can further be used in text classification tasks.
4. Anomaly and outlier detection: It can also be used in detecting anomalies as well as outliers for further analysis and investigation.

5. Risk assessment: Agglomerative or divisive clustering can be used to group different risk factors in order to better understand the overall risk of a portfolio.
6. Network analysis: Hierarchical clustering can be used to group nodes in a network based on their connections, which can then be used to better understand network structures.

STOP TO CONSIDER

Hierarchical clustering algorithms are used in may disciplines such as biology, image analysis and the social sciences to explore and recognize patterns in datasets.

3.3.1 BIRCH

BIRCH (**B**alanced **I**terative **R**educing and **C**lustering using **H**ierarchies) algorithm was proposed by Zhang, Ramakrishnan and Livny. It is a hierarchical agglomerative clustering algorithm. BIRCH is designed for clustering large amount of numeric data using a limited amount of memory. It is a multiphase hierarchical clustering algorithm where in initial phase it integrates hierarchical clustering called micro-clustering and other clustering methods are used in later phase called macro-clustering. BIRCH at any stage has a small set of subclusters and for the current stage it merges the subclusters based on some criteria. It does not maintains the individual objects of the subcluster. BIRCH handles this task by maintaining a set of Cluster Features(CF) of the subclusters. The decision to merge two subclusters is taken by the information provided by the CF's of the respective subclusters. The CF's of different subclusters are maintained in a tree called the CF tree and it requires only one pass to construct the CF tree as well as it's subsequent stages. The last of the CF tree which is optional requires another pass.

Now let's study in details about the Clustering Features and the CF tree.

Cluster Feature Vector: Let C be a cluster containing the data objects $O_1, O_2, O_3 \ldots\ldots O_n$. It is a 3-D vector summarizing information about clusters of objects. It is defines as,CF = (n, ls, ss) where n is the number of data objects in C, l is the linear sum of the data objects and ss is the square sum of the data objects in C. In other words:

$|C| = n;$

$\Sigma O_i = ls$

$i \in C$

$\Sigma O^2_i = ss$

$i \in C$

As the objects are multidimensional, the summations and exponentiation in the above expressions are dealt with component-wise. The dimensions of ls and ss are the same as that of the objects in C. For example, if $O_1(x_{11}, x_{12}, x_{13})$, $O_2(x_{21}, x_{22}, x_{23}), \ldots\ldots, O_n(x_{n1}, x_{n2}, x_{n3})$ then

$ls = (ls_{(1)}, ls_{(2)}, ls_{(3)})$, then

$ls_{(1)} = x_{11} + x_{21} + \ldots\ldots\ldots\ldots + x_{n1}$

$ls_{(2)} = x_{12} + x_{22} + \ldots\ldots\ldots\ldots + x_{n2}$

$ls_{(3)} = x_{13} + x_{23} + \ldots\ldots\ldots\ldots + x_{n3}$

$ss = (ss_{(1)}, ss_{(2)}, ss_{(3)})$ such that

$ss_{(1)} = x_{11}^2 + x_{21}^2 + \ldots\ldots\ldots\ldots + x_{n1}^2$

$ss_{(2)} = x_{12}^2 + x_{22}^2 + \ldots\ldots\ldots\ldots + x_{n2}^2$

$ss_{(3)} = x_{13}^2 + x_{23}^2 + \ldots\ldots\ldots\ldots + x_{n3}^2$

BIRCH assumes that the distance functions between clusters are so defined that they can be expressed in terms of the parameters encoded in the CF vectors.

Additive Properties of Cluster Features

Let us assume two cluster features $CF_1 = (n_1, ls_1, ss_1)$ and $CF_2 = (n_2, ls_2, ss_2)$ such that $C_1 \cap C_2 = \emptyset$. Then by merging these two disjoint clusters, the CF vector of the merged cluster will be $(n_1 + n_2, ls_1 + ls_2, ss_1 + ss_2)$.

If the CF vectors of a cluster are given then the following metric information can be calculated.

Centroid

The centroid is defined as an object for a cluster $C = \{O_1, O_2, \ldots\ldots\ldots O_n\}$

$$O_{centroid} = \frac{\sum_{i=1}^{n} O_i}{n!} = \frac{l}{s}$$

Radius

The radius R of a cluster $C = \{O_1, O_2, \ldots\ldots\ldots O_n\}$ is defined as

$$R = \left[\frac{\sum_{i=1}^{n}(O_i - O_{centroid})^2}{2}\right]^{1/2} = \left[\frac{ss - 2\frac{(ls)^2}{n} + \frac{(ls)^2}{n^2}}{n}\right]$$

Diameter

The diameter of a cluster $C = \{O_1, O_2, \ldots\ldots\ldots O_n\}$ is defined as

$$D = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j=1}^{n}(O_i - O_j)^2}{n(n-1)}}$$

Euclidean Intercluster Distance D_o

The Euclidean intercluster distance of two clusters C_1 and C_2 with centroids $O_{centroid1}$ and $O_{centroid2}$ respectively is defined as

$D_0 (C_1,C_2) = [\sum_{i=1} (O^i_{centroid1} - O^i_{centroid2})^2]^{1/2}$

where the subscript denotes the components of a vector.

Manhattan Intercluster DistanceD_1

The Manhattan intercluster distance of two clusters C_1 and C_2 with centroids $O_{centroid1}$ and $O_{centroid2}$ respectively is defined as

$D_1 (C_1,C_2) = [\sum_i | O^i_{centroid1} - O^i_{centroid2}|]$

Average Intercluster DistanceD_2

$D_2 (C_1,C_2) = [\frac{1}{n_1 n_2} \sum_{i \in C_1} \sum_{i \in C_2} (O_i - O_j)^2]$

Average Intracluster Distance

$D_3 (C_1 , C_2) =$

$[\frac{n(n-1)x^2}{(n_1+n_2)(n_1+n_2-1)} \sum^n_{i,j \in C_1 \cup C_2} (o_i - O_j)^2]$

Now let’s study about the four phases of the BIRCH clustering process.

Phase I Construction of a CF tree

A CF tree is constructed to store the CF’s for hierarchical clustering. In a CF tree, for a fixed number of clusters the CF’s are stored at each leaf node. The sum of CF’s of it’s child node is stored at each non-leaf node A CF tree is a height balanced tree with two important parameters: Branching factor B and the Diameter threshold denoted by T.Each non-leaf node contains atmost B entries of the form [CF_i, $child_i$] where $child_i$is a pointer to it’s child node and CF_i is the cluster feature of the subcluster represented by this child. On the other hand a leaf node contains B entries each of the form $[CF_i]$. The two pointers 'prev' and 'next' in each leaf node can be used to chain all leaf nodes together for an efficient scan. A leaf node represents a cluster made up of all subclusters represented by it's entries. The threshold value T is given such that all the entries in a leaf node must satisfy the threshold requirements. The threshold requires each diameter of the subcluster must be less than T. A CF tree can be incremented dynamically as new data objects are inserted.

The process of insertion of new data objects in a CF tree is similar to B and B^+ trees. It is assumed that the readers have the concept of the above trees. Let's understand the stages involved in the insertion process with the help of an example.

Let $CF_0 = [1, O, O^2]$

Identifying The Appropriate Leaf

While inserting a node O, the process starts from the root and it moves recursively downwards by selecting at every step the closest child node according to the chosen distance metric. Using CF_1 and CF_0 the distance between the ith subcluster and the object O is calculated to determine the cluster that is closest to O and hence O moves to the corresponding child node.

Modifying The Leaf Node

It selects the closest leaf entry say L_i when the leaf node is reached and is represented by it's CF. Now either O is absorbed into some subcluster in the leaf node or has a separate entry into this node.

Absorbing O in L_i

Let L_i represent a subcluster whose diameter is less than the threshold T. O is introduced to L_i only if the inclusion O to the subcluster does not violate the threshold condition; that is the diameter of the new cluster still remains below T. Otherwise O is introduced as a separate entry into the leaf node.

Introduce O In The Leaf Node

A new entry corresponding to O is added to the leaf and it represents a subcluster with the single element O. The step of introducing a node is only possible if there is space on the leaf node. When the leaf node is full, the new entry leads to the splitting of this leaf node.

Splitting Of The Leaf Node

The splitting of the leaf node leads to form two leaf nodes. The criteria for node splitting is done by choosing the farthest pair of entries as seeds in the leaf node and finally redistributing the remaining entries based on the closest criteria.

Modifying The Path To The Leaf

- When O is absorbed in L_i , the CF vector of L_i is modified with a new value. This in turn, will affect the cluster features of all the nodes from L_i to the root.
- Similarly if O is added as a separate entry on L_i, but without splitting L_i ,then the CF vector of the parent node of L_i will be updated and all the other nodes in the path will reflect this change.
- If there is a node split, there is a new entry at the parent node of L_i, and this entry consists of the CF vector together with a pointer to the new leaf node. This process may propagate upwards, in case the parent node of L_i does not have enough space to accommodate the new entry. It may be necessary to split the parent as well and so on up to the root. If the root is split, the height of the tree increases by one.

Merging Refinement

It may so happen that there is a leaf split and the propagation of this split stops at some non-leaf node, N_j. In case of such a split, N_j can accommodate the additional entry. Now we need to scan N_j to find the two closest entries. If they are not the pair corresponding to the split, we try to merge them and corresponding two child nodes. The merging is splitted again by redistributing if there are more entries in the two child node to be accommodated in a merge. During such re-splitting, in case one seed attracts enough merged entries to fill a page, we just put the rest of the entries with the other seed.

Let's understand Phase I Construction of CF tree with the help of an example depicted in Figure 4.

Let us assume there are 8 objects in order and the branching factor be 3. The first three objects O_1,O_2 and O_3 are inserted as initially the CF tree consist of just one node. As soon as the O_4 is inserted, the node is split to two leaf nodes corresponding to the farthest pair of objects O_1 and O_4. The other objects are assigned to the leaf nodes based on the closest distance condition. As the second object is closer to O_1, it is in the first leaf node. The details of the two subclusters corresponding to the leaf nodes is maintained by the root node. As the distance of the next new object $O_5$is beyond the threshold T from any of the subclusters, a new child node of the root node is created. After this, the details at the root node and the new leaf node are updated. The object O_6 is now allocated to the third leaf node according to the closest distance criterion. As the objects O_7 and O_8 are allocated to the first leaf node and B=3, so the node is now splitted into two leaf nodes.

Thus in Phase I we can see that a CF tree is created by scanning the data set record by record to form a CF tree. If main memory cannot accommodate the tree , the leaf nodes are reinserted to make the tree smaller. This method requires just one pass scan of the database if B and T are properly selected.

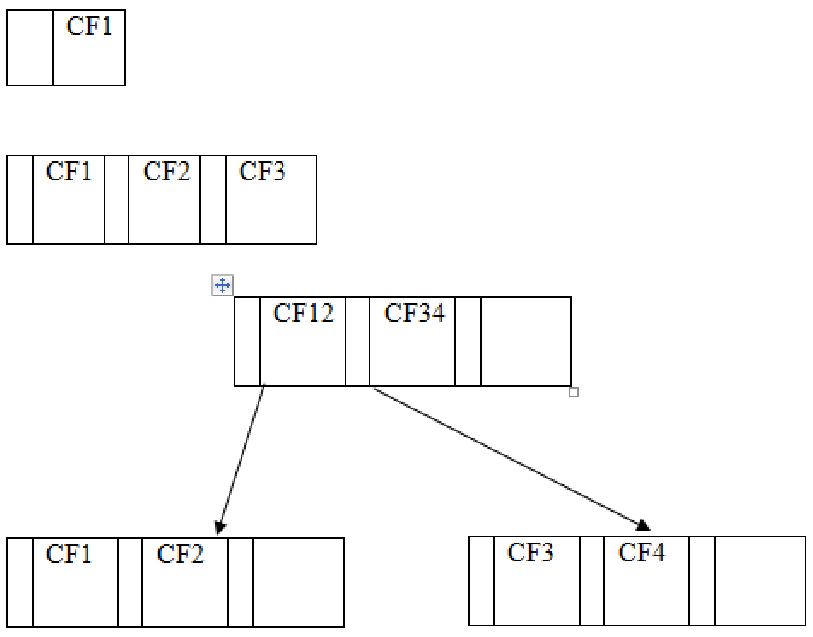


Figure 5 (a) CF Tree Construction Process O_1, O_2, O_3 and O_4 are inserted

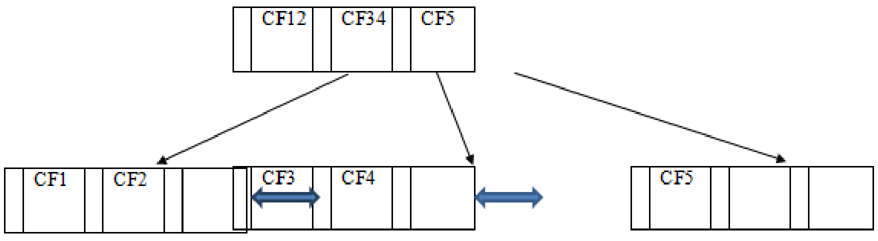


Figure 5 (b) CF Tree after inserting O_5

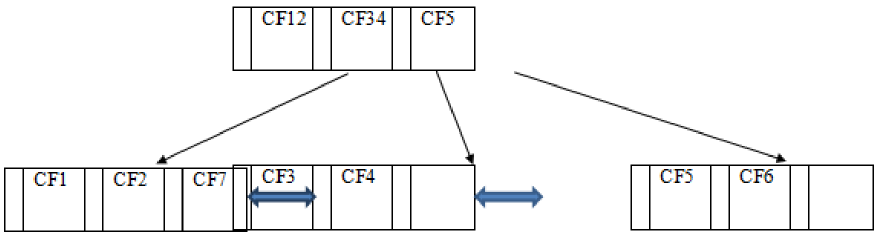


Figure 5 (c) CF Tree after O_6, O_7 are inserted. The root node and left-most child nodes are full.

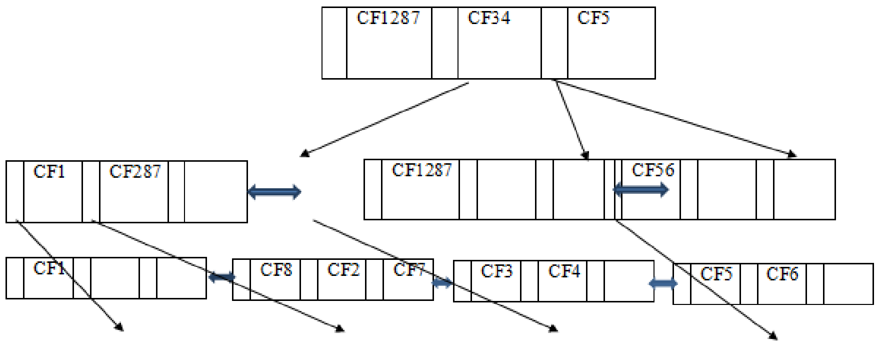


Figure 5 (d) CF Tree after O_8 is inserted. Nodes split and the tree expand.

Phase II Condensation of CF Tree

In phase I we can see that the size of the CF tree is dependent on B and T. A smaller T generates a larger tree. A set of clusters whose diameter does not exceed the threshold T is represented at the leaf node and the centroids of any two subclusters are separated by a distance T. At most, B number of the closest subclusters at one level are merged to represent a parent node at the next higher level. If there are too many subclusters at the leaf level, then T becomes too small. Thus for a large number of points, the CF tree might be too deep.

In Phase II the threshold is redefined and the CF tree is processed according to the new threshold so that the size of the tree becomes manageable. Phase II also aims for removing the isolated subclusters as outliers during the process of condensing. After removing the outliers, a CF tree is rebuilt by scanning the leaf entries of the initial CF tree.

Phase III Hierarchical Agglomerative Clustering

Phase III starts with the set of subclusters at the leaf node. In order to obtain the set of user specified clusters any one of the conventional hierarchical clustering techniques is utilized. The user specifies the target number of clusters. In this phase, the clustering technique is employed directly to the subclusters which are represented by their CF vectors. The distance metric, which is discussed in the earlier section, can be calculated from the CF vectors. The advantage of this method is that the user either specifies the desired number of clusters or the threshold diameter. This phase doesn’t require any database scan.

Phase IV Classification

The centroids of the clusters that were produced by Phase III are used as seeds in Phase IV and are used to redistribute the data objects to I i.e, the closest seed to obtain a set of new clusters. This allows the objects to migrate from one cluster to the other and it ensures that multiple copies of the same object presented are included in the same clusters. The CF building algorithm depend on how the data objects are being presented and it is quite possible that the two copies of the same object presented at different instances of the sequence may land up at different nodes. CF tree is one of the novel algorithms for data mining.

Advantages

1. It is local in that each clustering decision is made without scanning all data points and existing clusters.

2. It exploits the observation that the data space is not usually uniformly occupied, and not every data point is equally important.

3. It uses available memory to derive the finest possible sub-clusters while minimizing I/O costs.

4. It is also an incremental method that does not require the whole data set in advance.

Disadvantages

1. It can only process metric attributes.

CHECK YOUR PROGRESS

1. A ______ is a tree-like structure that explains the relationship between all the data points in the system.

2. BIRCH is a hierarchical ________ clustering algorithm.

3. BIRCH maintains a set of ___________ of the subclusters.

4.________ can only process metric attributes.

5. State an advantage and a disadvantage of BIRCH algorithm.

6. State two applications of hierarchical algorithms.

3.3.2 CURE

CURE (**C**lustering **U**sing **R**epresentatives) is another hierarchical clustering which adopts an agglomerative scheme. For each subcluster in CURE, it maintains a set of representative points. The representative points are reasonably smaller in number to avoid the inefficiency of the all-points strategy.The representative points are so well scattered within the subcluster that it represents the whole subcluster. As the subcluster is spherical in shape, the centroid is the best representative of the shape. But practically, it cannot be guaranteed that the clusters are spherical in shape.

Now let's discuss the working of the algorithm. The algorithm begins with every single data object as a cluster and the object itself is the sole representative of the corresponding cluster.

At any stage of the algorithm, we have a set of with a set of representative points associated with it. The distance between two subclusters is the smallest pair wise distance between their representative points. For every subcluster C, it's nearest subcluster $C_{nearest}$ is computed at this stage and is defined as:

$D_{closest}(C) = \text{Distance} (C, C_{nearest})$

The subclusters are arranged in increasing order of $D_{closest}(C)$. As hierarchical clustering works by merging the closest pair of subclusters, the subcluster C corresponding to the smallest value of $D_{closest}(C)$ is the candidate subcluster to be merged with it's nearest subcluster $C_{nearest}$ for the next stage. After merging the clusters, a new set of representative points are computed for the merged cluster. The task of merging clusters continues till the specified number of clusters are obtained.

Now the problem is how to compute the set of representative points of the new subcluster. It is not always possible for a set of representative points of individual subcluster to represent the well-scattered set of points of the merged cluster.So the method to recompute the representative points of the newly formed cluster is as follows. It begins with the centroid of the new cluster and finds the farthest data object from the centroid. This is the first representative point. In the next iterations, a point in the subcluster is selected in such a way that it is farthest from the previously chosen representative points. The points are then shrunk towards the centroid by a fraction α. This is done to get rid of the outlier object of being the scattered representative. At every stage, a heap data structure is maintained to determine the closest pair of subclusters. It also maintains a k-d tree for it's efficient implementation.

Advantages

1. It can be used for both spherical and non-spherical type of clusters.

2. Good execution time in case of large databases due to the use of random sampling and partitioning techniques.

3. Works well even when the database contains outliers.

Disadvantages

1. There is no provision for deciding the optimal number of clusters.

CHECK YOUR PROGRESS

7. For each subcluster in CURE, it maintains a set of __________ points.

8.______ can be used for both spherical and non-spherical type of clusters.

9. What is the disadvantage of CURE algorithm?

3.4 Summing Up

1.Hierarchical algorithm creates a hierarchical decomposition of the database by grouping the data points that are similar.

2.The data points that are close to each other are considered to be similar while the other data points are considered to be dissimilar and are placed in other clusters.

3. In this way clustering is continued until all the data points are grouped into clusters based on their similarity

4. A dendrogram is a tree-like structure that explains the relationship between all the data points in the system

5. This dendrogram can be represented in two ways: agglomerative and divisive approach.

6. The advantage of hierarchical algorithms are that the number of clusters need not be specified, easy to understand and interpret, handle large datasets without being computationally expensive and time consuming.

7. The main disadvantage of hierarchical algorithm is that it is sensitive to noise and outliers.

8. The applications of hierarchical algorithm include customer and image segmentation, text analysis, anomaly and outlier detection and many more.

9. BIRCH is a hierarchical agglomerative clustering algorithm.

10. BIRCH at any stage has a small set of subclusters and for the current stage it merges the subclusters based on some criteria.

11. BIRCH handles this task by maintaining a set of Cluster Features(CF) of the subclusters.

12. The CF's of different subclusters are maintained in a tree called the CF tree and it requires only one pass to construct the CF tree as well as it's subsequent stages.

13. The phase I of constructing a CF tree consists of identifying the appropriate leaf, modifying the leaf node, splitting the leaf node, modifying the path to the leaf and finally the merging refinement.

14. Phase II, III and IV consists of condensation of CF tree, hierarchical agglomerative clustering and classification respectively.

15. One of the advantage of BIRCH algorithm is that it uses available memory to derive the finest possible sub-clusters while minimizing I/O costs and also it is an incremental method that does not require the whole data set in advance.

16. CURE is another agglomerative clustering algorithm that it maintains a set of representative points.

17. The representative points are so well scattered within the subcluster that it represents the whole subcluster.

18. One of the advantage of CURE algorithm is that it can handle both spherical and non-spherical type of clusters.

3.5 Answers to Check Your Progress

1. Dendrogram

2. Agglomerative

3. Cluster Features(CF)

4. BIRCH

5.The advantage of BIRCH algorithm is that it exploits the observation that the data space is not usually uniformly occupied, and not every data point is equally important. The disadvantage of BIRCH algorithm is that it can process only metric attributes.

6. The two applications of hierarchical algorithms are:

i) Hierarchical Clustering can be used to group customers in different clusters depending upon the buying patterns, demographic information. This can be of great help in studying the customer behavior for target making campaigns.

ii) Network analysis: Hierarchical clustering can be used to group nodes in a network based on their connections, which can then be used to better understand network structures.

7. Representative

8. CURE

9. The disadvantage of CURE algorithm is that there is no provision for deciding the optimal number of clusters.

3.6 Possible Questions

1. What is a hierarchical clustering algorithm?

2. What are the two ways in which a dendrogram can be represented?

3. State the difference between agglomerative and divisive approach with the help of an example.

4. State the advantages and disadvantages of hierarchical clustering algorithm.

5. What are the different applications of hierarchical clustering algorithm?

6. State the working of the BIRCH algorithm.

7. What is Cluster Feature (CF) vector?

8. Explain how a CF tree is constructed.

9. State the advantages and disadvantages of BIRCH algorithm.

10.State the working of the CURE algorithm.

3.7 References and Suggested Readings

[1]Pujari, A. K. (2001). *Data mining techniques*. Universities press

[2] Mining, W. I. D. (2006). *Introduction to data mining* (pp. 2-12). New Jersey: Pearson Education, Inc.

[3] https://builtin.com/machine-learning/agglomerative-clustering

[4]https://towardsdatascience.com/hierarchical-clustering-explained-e59b13846da8

[5]https://www.geeksforgeeks.org/agglomerative-methods-in-machine-learning/

[6]https://codinginfinite.com/wp-content/uploads/2022/12/Screenshot-from-2022-12-05-19-16-46.png

[7]https://codinginfinite.com/hierarchical-clustering-applications-advantages-and-disadvantages/

[8] https://www.geeksforgeeks.org/ml-birch-clustering/

[9] https://www.javatpoint.com/birch-in-data-mining

---×---

UNIT: 4
CATEGORY AND DENSITY BASED ALGORITHMS

Unit Structure:

4.1 Introduction

In this unit we will study about the category and density-based clustering algorithms. Categorical clustering deals with categorical data whereas density-based algorithms can be used for arbitrary shaped clusters. In categorical clustering, some common properties of the cluster or sub-clusters are used to join or split two clusters depending on some user specified values for common properties. In density-based algorithm, density is measured from a particular point in the dataset or database by counting the number of points within a specific radius, including the point located at the center. We will also study the underlying principle of the different algorithms that belong to the above categories.

4.2 Unit Objectives

After going through this unit you will be able to:

- Understand category-based clustering technique
- Understand density-based clustering technique
- Understand the types of category and density-based algorithms
- Understand the underlying principle of the category-based algorithms: ROCK and CACTUS
- Understand the underlying principle of the density-based algorithm DBSCAN

4.3 Category-based Algorithms

As the name suggest, Category based algorithms deals with categorical data. So before discussing Categorical based algorithms we should have the idea about the categorical data. Categorical data can be defined as numerical or non- numerical data which can be grouped or categorized depending on some common characteristics among them. The characteristics may be like designation of employees, weight and height of students, colours of cars etc. The values for numerical categorical data includes a range of values rather than a fixed value. So for clustering these type of data, distance functions are not applicable. Categorical based clustering often joins or splits sub-clusters based on the similarity and dissimilarity respectively based on some user specified values.
Let's understand the categorical-based clustering technique elaborately with the help of two algorithms ROCK and CACTUS.

4.3.1 ROCK

ROCK (Robust hierarchical-Clustering with Links) is an adaptation of agglomerative hierarchical clustering algorithm. Agglomerative hierarchical clustering normally uses distance-based representatives to determine similarity between clusters. ROCK makes use of links for defining similarity. The number of common neighbours in the dataset represents the number of links between two tuples. For each tuple in it's own cluster, the clusters that are closest are merged until the required number of clusters are obtained. ROCK randomly selects a sample for clustering drawn from the dataset rather than clustering the whole dataset. Later this sample is used for partitioning the entire dataset based on the clusters from the sample. Let's learn some definitions first to understand the ROCK algorithm.

NEIGHBOURS

Neighbours of an object are those objects that are similar or share the common characteristics with the object. Let $sim(O_i, O_j)$ be a similarity function that represents the similarity or captures the closeness between the pair of objects O_i and O_j. The range of values that function sim can take lies between 0 and 1. When a threshold θ is given such that it's value lies between 0 and 1 then the pair of objects O_i and $O_j$are defined as neighbours if

$$sim(O_i, O_j) \geq \theta$$

LINKS

The number of common neighbours between O_i and O_j is defined as the $link(O_i,O_j)$ between the pair of the objects O_i and O_j. If the $link(O_i,O_j)$ is large, then it is more probable that O_i and O_j belong to the same cluster. ROCK maximizes the sum of $link(O_q,O_r)$for the pair of objects O_q and O_r belonging to a single cluster and at the same time minimizes the sum of the $link(O_q,O_r)$ for the pair of objects belonging to different clusters.

LINK BETWEEN CLUSTERS

For two clusters, C_i and $C_j$the $link[C_i, C_j]$ can be defined as

$$link[C_i, C_j] = \sum_{Oq \in Ci, Or \in Cj} link(O_q, O_r)$$

GOODNESS MEASURE

The goodness measure $g(C_i, C_j)$ for merging two clusters C_i , C_j is defined as

[illegible]

$$g(C_i,C_j) = \frac{link[C_i,C_j]}{(n_i+n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}}$$

Now let us explain the different definitions with the help of an example.

Let us consider a domain D = {1,2,3,4,5}
O_1 = {1,2,3}
O_2 = {1,2,4,5}
O_3 = {2,4,5}
O_4 = {3,4,5}

Let us draw a table with the Boolean values where the presence of an attribute is represented as 1 or 0.

Object	**1**	**2**	**3**	**4**	**5**
O_1	1	1	1	0	0
O_2	1	1	0	1	1
O_3	0	1	0	1	1
O_4	0	0	10	1	1

Table 1: Object representation

Now we will calculate the *sim()* value of *O_1*with respect to other objects.

If we set the threshold value θ = 0.4, then *O_1* and *O_2*are neighbours.
If *sim() = 1* for the two objects *O_1* and *O_2*then they are identical.
If *sim() = 0*for the two objects *O_1* and *O_2*have no common characteristics.

Let us now generate a table with all the similarity values of the object pairs as follows:

	O_1	**O_2**	**O_3**	**O_4**
O_1	1	0.4	0.2	0.2
O_2	0.4	1	0.75	0.4
O_3	0.2	0.75	1	0.5
O_4	0.2	0.4	0.5	1

Table 2: sim() values

For θ = 0.4 we find that

neighbour (O_1) = {O_1, O_2}
neighbour (O_2) = {O_1, O_2,O_3, O_4}
neighbour (O_3) = {O_2, O_3, O_4}
neighbour (O_2) = {O_2,O_3, O_4}

Now *link(O_i, O_j)* can be computed as

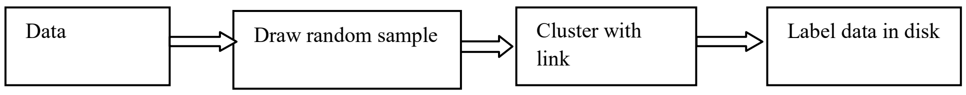
link(O_i, O_j) = | neighbour(O_i) ∩ neighbour(O_j)|

We can also generate the link() values as shown in the table below

	O_1	**O_2**	**O_3**	**O_4**
O_1	-	2	1	1
O_2	2	-	4	4
O_3	1	4	-	3
O_4	1	4	3	-

The ROCK algorithm can be divided into three steps as given below:

Here in the steps we can see in the first step it draws the random samples from the data in the space then in the second step, it follows the procedures of finding links between the data points which are randomly collected from the dataset and by employing those links.



i) Draw a random sample from the dataset: In the first step, it draws random samples from the data in the space. The initial sampling is used to form clusters and then the remaining data objects on the disk are assigned to these clusters.

ii) Perform a hierarchical clustering algorithm: A hierarchical agglomerative clustering technique is applied to the random sample selected in this step. Each data object is labelled as an individual cluster. After this merging of clusters is conducted based on the highest similarity using the different similarity measures. The merging process is continued until we get the defined number of clusters or no more links remains between the clusters.

iii) Label the data in the disk: In this step, we assign the remaining data objects to the generated clusters. This is accomplished by selecting a random sample L_i for each cluster C_4 then we assign the remaining data objects P to the cluster for which it has strongest linkage with L_i.

Now let's go through the ROCK algorithm:

ROCK

```
Input: (S,k) where S is the set of n sampled objects to be clustered and
k is the number of desired clusters.
Initially the link for each pair of objects in S is computed.
For each s ∈ S, q[s] maintains a local heap structure of objects linked
to s in decreasing order.
A global heap, Q is also maintained to indicate the number of clusters
that are created so far out of S.
while(size(Q)>k)) do
  select a cluster u from Q and select the nearest element v from q[u]
  delete v from Q
  w = merge(u,v)
  for each x ∈ q[u] ∪ q[v] do
     link[x,w] = link[x,u] + link[x,v]
      delete u from q[x]
      delete v from q[x]
       insert w to q[x] and compute g(x,w)
        insert x to q[w] and g(x,w)
    end
    insert(Q,w,q[w])
end
```

```
Procedure for computing links
begin
    compute nbrlist[O_i] for every point i in S
    set link[O_i,O_j] to zero for all I,j
         for i=1 to n do
            N = nbrlist[O_i]
            for j =1 to |N| -1 do
              for l = j+1 to |N| do
```

```
                link[N[$O_j$],N[$O_l$]] = link[N[$O_i$],N[$O_j$]] +1
end
```

4.3.2 CACTUS

CACTUS – Clustering Categorical Data Using Summaries-was developed by Ganti, Gehrke and Ramakrishnan. It is a fast summarization-based algorithm for clustering categorical data. Categorical domain data are always characterized by having a small number of attribute values. The idea behind this algorithm is that a summary of the entire dataset is sufficient to compute a set of candidate clusters which can then be validated to determine the actual set of clusters. The key ideas of the algorithm is that as the summary information easily fits into the main memory, it can be constructed efficiently in a single scan of the dataset. Before understanding the underlying principle of the CACTUS , let us define some terms that would help us to understand the algorithm better. Let us consider two different attribute values of two different attributes in the database. Let a_i be of attribute type A and a_j of attribute type B. There may be tuples where a_i and a_j co-occur. The support of these two tuples in the database is the proportion of tuples in which they appear together. If this support exceeds a pre-specified value, we say that these values are strongly connected. Let D_A denote domain of attribute A and s denotes the support value.

Strongly Connected Attribute Values

Let $a_i \in D_A$, $a_j \in D_B$ and $\alpha > 1$. The attribute values a_i and a_j are strongly connected with respect to domain D if

$$s_D(a_i, a_j) > \frac{\alpha|D|}{|D_A||D_B|}$$

The function $\sigma^*(a_i, a_j) = \begin{cases} s_D(a_i, a_j) & \text{if } a_i \text{ and } a_j \text{ are strongly connected} \\ 0, & \text{otherwise} \end{cases}$

The above definition can be extended to define strongly connected value sets. Let a_i be an attribute value and S a set of attribute values of another attribute type. We say that a_i is strongly connected to S if it is strongly connected to every element of S. Similarly two attribute value sets S and R are said to be strongly connected if each value a in S is strongly connected to R and each value in b in R is strongly connected to S. using this concept we define a cluster.

Cluster

Let C_i be the attribute value set of ith attribute; a subset of the domain D_i, such that $|C_i| > 1$. Then C= $<C_1, \ldots\ldots\ldots C_2>$ is said to be a cluster over the database of n attributes if the following three conditions are satisfied:

1. For all $i,j \in \{1,\ldots.n\}$, $i \neq j$, C_i and C_j are strongly connected.
2. For all $i, j \in \{1,\ldots.n\}$, $i \neq j$, there exists no $C'_i \supset C_i$ such that for all $j \neq i$, C'_i and C_j are strongly connected.
3. $S_D(C) \geq \frac{\alpha|D|x|C_1|x\ldots\ldots.x|C_n|}{|D_1|x|D_2|x\ldots..x|D_n|}$

k-Cluster

A cluster on k-attributes is called a k-cluster. It is also known as subspace cluster.

Cluster Projection

A cluster on a database is called a cluster projection if it is a projection of the original database D with respect to the attributes present in the projection operation. The cluster C_i discussed aboveis called the cluster projection of C on attribute A_1.

Similarity Measure of Elements of One Attribute

The similarity measure between two values of an attribute say a_1 and a_2 belonging to domain D_i with respect to another attribute A_j, as the number of elements of D_j that are strongly connected to both a_1 and a_2

$\gamma^j (a_1, a_2) = |\{x \in D_j : \sigma_D^*(a_1 , x) > 0 \text{ and } \sigma_D^*(a_2 , x) > 0\}|$

Inter-Attribute Summary

Let there be a set of categorical attributes $A_1 \ldots\ldots\ldots A_n$ with domains $D_1 \ldots\ldots D_n$ respectively and let D be a dataset. The inter-attribute summary Σ_{IJ} is defined as

$\Sigma_{IJ} = \{\Sigma_{ij} | \, i,j \in \{1\ldots\ldots n\}$ and $i \neq j\}$

where $\Sigma_{ij} = \{(a_i, a_j, \sigma^*_D(a_i, a_j)) \, | a_i \in D_i, a_j \in D_j$ and $, \sigma^*_D(a_i, a_j)) > 0 \}$

Intra-Attribute Summary

The intra-attribute Σ_{ij} is defined as

$\Sigma_{IJ} = \{\Sigma_{ij} | \, i,j \in \{1\ldots\ldots n\}$ and $i \neq j\}$

where $\Sigma^j_{ij} = \{(a_i, a_j, \gamma^j (a_{i1}, a_{i2})) \mid a_{i1}, a_{i2} \in D_i$ and $\gamma^j (a_{i1}, a_{i2}) > 0\}$

CACTUS
The CACTUS algorithm starts by first identifying the cluster projections on an attribute for the two clusters involving this attribute. After this an intersecting set is generated to represent the cluster projection on this attribute for n-cluster. Once this is done, they are synthesized to get clusters of database. Thus the steps in CACTUS can be summarized as:

1. Finding cluster projections CPi[i,j] on a given attribute A_i with respect to another attribute A_j in a 2-cluster involving these two attributes. This is done for all attributes A_i.
2. Intersecting these CPi[i,j] for a fixed A_i to get $CP_i[1,2\ldots..k]$; that is , a cluster projection of A_i with respect to all the attributes. It is also done for all A_i.
3. Synthesizing the set of CP_i [1,2……..,k] for all I, to get the clusters.

CACTUS Algorithm
The three steps of CACTUS algorithm is described below in the following steps:

i) Summarization: In this step the summary information is computed from the dataset. The Inter and Intra attribute summary fits easily into main memory.

a) Inter-attribute Summaries: Initially for each pair(a_i, a_j) D_i x D_j where $i \neq j$ set a counter to 0. After this a scan of the dataset is done every time, for each pair such that the pair is found in a tuple in the database. The support of each pair is calculated. After the scan, compute and reset the counters of those whose $\sigma^* E[\sigma_D(a_i,a_j)]$. Now those pairs are stored.

b) Intra-attribute Summaries: In order to find a tuple pair say (T1,T2) scan the dataset and find the domain such that T1.a is strongly connected with T1.b and T2.a is strongly connected with T2.b. This is computed when it is needed only as this operation is very fast.

ii) Clustering: In clustering step the summary information is used to discover a set of candidate clusters. This are the two steps involved in the clustering phase:

a) Analyze each attribute to compute all cluster-projection.

b) Synthesise candidate clusters on sets of attributes from the cluster projections on individual attributes.

iii) Validation: This step determines the actual set of clusters from the set of candidate clusters. Validation is done by checking in inter attribute summaries if pair in the cluster is strongly connected. The result is validated by checking if they are all strongly connected.

CHECK YOUR PROGRESS

1. ROCK is an adaptation of_______________ hierarchical algorithm.

2. In ROCK, the number of common __________ in the dataset represents the number of links between two tuples.

3. Define Neighbour, Similarity function and link.

4. _________ is a fast summarization-based algorithm for clustering categorical data

5.Define cluster and cluster projection.

4.4 Density Based Clustering Algorithm

Clustering algorithms always have the problem of dealing with sparsity of data. This can be handled with the use of Density based clustering algorithm as it can discover arbitrary shaped clusters. Density based clustering works by identifying dense cluster of points allowing it to learn clusters of arbitrary shape and identify outliers in the data. The Density based clustering algorithm works by detecting areas where points are concentrated and where they are separated by areas that are empty or sparse. The points that lay outside the cluster are referred to as noise. We will be understanding this concept with the DBSCAN (Density Based Spatial Clustering of Applications of Noise) algorithm discussed in the next section.

4.4.1 DBSCAN

DBSCAN uses a density based notion of clusters to discover clusters of arbitrary shape. For each object of a cluster in DBSCAN, the neighborhood of a given radius has to contain atleast a minimum number of data objects. In other words, the density of the neighborhood must exceed the minimum threshold. The most important parameter here is the distance function of the data objects. The following concepts are necessary to understand the concept of the DBSCAN algorithm.

ε-Neighbourhood of an Object

For a given non-negative value ε, the ε-neighbourhood of an object O_i denoted by $N\varepsilon\ (O_i)$ is defined by

$N\varepsilon\ (O_i) = \{O_j \in D \mid d(O_i, O_j) \leq \varepsilon\}$

Core Object

An object is said to be a Core Object if $|N\varepsilon\ (O)| \geq$ MinPts. A core object is an object which has a neighbourhood of user-specified minimum density.

Directly-Density-Reachable

An object O_i is directly-density-reachable from an object O_j with respect to εand Minpts, if $O_j$is a core object and O_i is in it'sε-Neighbourhood

1. $O_i \in N\varepsilon (O_j)$
2. $|N_\varepsilon(O_j) \leq$ MinPts (In other words, $O_j$is a core object)

Density-Reachable

An object O_i is density reachable from an object O_j with respect to ε and MinPts in D if there is a chain of object O_1, O_2 , ...O_n such that $O_i = O_j$, $O_n = O_i$ such that $O_e \in$ D and O_{e+1} is directly density-reachable from O_e with respect to ε and MinPts in D.

Noise

Let C_1,C_2,..........C_n be the clusters in the domain D with respect to ε and MinPts in D. Suppose if there are some elements N_1,N_2,………$N_n \in$ D which do not belongs to any of the cluster then Noise can be defined as

$N = \{O \in D | \forall i, O \notin C_i\}$

DBSCAN defines two kinds of objects: core and non-core objects. Non -core objects are either the border objects or noise objects. Border objects can be defined as the objects which are always density reachable from a core object whereas a non-core object is always not density reachable from other core objects. In DBSCAN finding a cluster is based on the fact that a cluster is uniquely determined by any of it's core objects. Each classified object has an associated cluster-id in which it is included. A noise object may also have a dummy cluster id associated with it. The ε-neighbourhood are already computed for both classified and noise objects. The objects that are not classified i.e, unclassified objects do not have a cluster id nor their neighbourhoods are computed initially. But as the algorithm proceeds, an unclassified object gets convert into a classified or a noise object.

STOP TO CONSIDER

DBSCAN algorithm was proposed in 1996. In 2014, the algorithm was awarded the 'Test of Time' award at the leading Data Mining conference, KDD.

DBSCAN algorithm starts with an unclassified object and a new cluster-id associated with it. The ε- neighbourhood of that object is examined to determine whether it is adequately dense or not. If the density does not exceed the threshold MinPts then it is considered as a noise object. Otherwise the objects that are within the ε-neighbourhood are retrieved and is put in a list of candidate objects. These objects may be either unclassified or noise. An important point to be kept in mind is that a classified object may reside in the neighbor. If the object is a noise object, then the current cluster id is associated with it. Again if the object is an unclassified object, then the current cluster id is associated with it and also it is put into the list of the candidate objects for which the ε-neighbourhoods are to be computed. This process is continued until the list containing the candidate object becomes empty. Thus one cluster is determined for the given cluster-id. The algorithm continues this process for all the unclassified objects and terminates only when all the objects are marked as either classified or noise.

The steps in the DBSCAN algorithm is as follows:
1 Arbitrarily select an object p
2 The ε-neighbourhood of p and mark it as a core object or non-core object.
3. If p is a core object, a cluster will be formed, then retrieve all the points that are density-reachable from p with respect to ε and MinPts.
4. If p is a noise object, no other points are density reachable from p and DBSCAN visits the next point of the database.
5. Repeat the steps 2-4 untill all the points have been processed.

After completion of these steps, all the points in the dataset will be classified as either belonging to a cluster or as a noise point.
The DBSCAN algorithm is given below:

DBSCAN ALGORITHM

```
Algorithm DBSCAN (D, ε ,MinPts)
Input: Database of objects D
do for all O ∈ D
    if O is unclassified
call function expand_cluster(O, D, ε,  MinPts)
end do
```

```
Function expand_cluster(O, D, ε, MinPts)
get the ε-neighbourhood of O as $N_ε$ (O)
  if |$N_ε$(O)| < MinPts,
        mark O as noise
        return
  else
        select a new cluster_id and mark all objects of Nε (O) with this cluster-id and put them into candidate-objects
        do while candidate-objects is not empty
            select an object from candidate-objects as current_object
            delete current-object from candidate-objects
            retrieve $N_ε$(current-objects)
            if |$N_ε$ (current-object)| ≥ MinPts
                select all objects in $N_ε$(current-object) not yet classified or marked as noise,
                mark all of the objects with cluster_id
                include the unclassified objects into candidate-objects
        end do

  return
```

Advantages:

a) It does not require one to specify the number of clusters
b) It can find arbitrarily shaped clusters and also find clusters surrounded by a different cluster
c) DBSCAN has a notion of noise
d) It requires just two parameters and is mostly insensitive to the ordering of the points in the database.

Disadvantages

a) Difficult to find an appropriate value for ε.
b) It cannot cluster datasets with large differences in densities well.

Applications

a) DBSCAN is used in satellite imagery.

b) It is also used in XRay crystallography

c) Anamoly detection in temperature is another application of DBSCAN

CHECK YOUR PROGRESS

6.Density based clustering algorithm as it can discover ___________ shaped clusters

7. Define core and noise

8. State one advantage and one disadvantage of DBSCAN algorithm

4.5 Summing Up

1. Categorical data can be defined as numerical or non- numerical data which can be grouped or categorized depending on some common characteristics among them.

2. Categorical based clustering often joins or splits sub-clusters based on the similarity and dissimilarity respectively based on some user specified values.

3. ROCK (Robust hierarchical-Clustering with Links) is an adaptation of agglomerative hierarchical clustering algorithm

4. ROCK makes use of links for defining similarity.

5. The number of common neighbours in the dataset represents the number of links between two tuples.

6. CACTUS is a fast summarization-based algorithm for clustering categorical data.

7. The CACTUS algorithm starts by first identifying the cluster projections on an attribute for the two clusters involving this attribute.

8. The second step in CACTUS is to generate an intersecting set to represent the cluster projection on this attribute for n-cluster. Once this is done, they are synthesized to get clusters of database.

9. Density based clustering algorithm can discover arbitrary shaped clusters.

10.The Density based clustering algorithm works by detecting areas where points are concentrated and where they are separated by areas that are empty or sparse.

11.DBSCAN uses a density based notion of clusters to discover clusters of arbitrary shape.

12. In DBSCAN finding a cluster is based on the fact that a cluster is uniquely determined by any of it's core objects.

4.6 Answers to Check Your Progress

1. Agglomerative

2. Neighbours

3. **Neighbour:**Neighbours of an object are those objects that are similar or share the common characteristics with the object.

Similarity Function:Let $sim(O_i, O_j)$ be a similarity function that represents the similarity or captures the closeness between the pair of objects O_i and O_j. The range of values that function sim can take lies between 0 and 1.

Link: The number of common neighbours between O_i and O_j is defined as the $link(O_i,O_j)$ between the pair of the objects O_i and O_j.

4. CACTUS

5. **Cluster:** Let C_i be the attribute value set of i^{th} attribute; a subset of the domain D_i, such that $|C_i| > 1$. Then $C = <C_1, \ldots\ldots\ldots C_2>$ is said to be a cluster over the database of n attributes if the following three conditions are satisfied:

1. For all $i,j \in \{1,\ldots.n\}$, $i \neq j$, C_i and $C_j$are strongly connected.
2. For all $i, j \in \{1,\ldots.n\}$, $i \neq j$, there exists no $C'_i \supset C_i$ such that for all $j \neq i$, $C'_i$and C_j are strongly connected.

3. $S_D(C) \geq \frac{\alpha|D|x|C_1|x.......x|C_n|}{|D_1|x|D_2|x.....x|D_n|}$

Cluster Projection:A cluster on a database is called a cluster projection if it is a projection of the original database D with respect to the attributes present in the projection operation. The cluster C_i discussed above is called the cluster projection of C on attribute A_1.

6. Arbitrary

7. **Core Object:**An object is said to be a Core Object if $|N\varepsilon\ (O)| \geq$ MinPts. A core object is an object which has a neighbourhood of user-specified minimum density.

Noise
Let C_1, $C_{2,}$...........C_n be the clusters in the domain D with respect to ε and MinPts in D. Suppose if there are some elements N_1, N_2,.........$N_n \in$ D which do not belongs to any of the cluster then Noise can be defined as

$N = \{O \in D | \forall i, O \notin C_i\}$

8. **Advantage**:It can find arbitrarily shaped clusters and also find clusters surrounded by a different cluster

Disadvantage: It cannot cluster datasets with large differences in densities well.

4.7 Possible Questions

1. What is categorical clustering?
2. What are the techniques for categorical clustering?
3. Explain the concept of a cluster used in ROCK. Compare this definition of cluster with that of DBSCAN.
4. What is a similarity function?
5. Describe the features of the ROCK algorithm.
6. Define the following:
   a. Inter-attribute summary
   b. Intra-attribute summary
c. Cluster projection
7. Describe the underlying principle of the CACTUS algorithm.

8. Define a link and a neighbor as defined in ROCK.
9. Describe the underlying principle of the DBSCAN
10. Mention two applications of DBSCAN

4.8 References and Suggested Readings

[1]Pujari, A. K. (2001). *Data mining techniques*. Universities press

[2] Mining, W. I. D. (2006). *Introduction to data mining* (pp. 2-12). New Jersey: Pearson Education, Inc.

[3] https://www.tutorialspoint.com/dbscan-clustering-in-ml-density-based-clustering

[4] https://www.geeksforgeeks.org/implementing-dbscan-algorithm-using-sklearn/

---×---

BLOCK- IV

UNIT:1
INTRODUCTION TO DECISION TREES

Unit Structure:

1.1 Introduction to Decision Trees

Decision trees are a widely used and effective tool in fields like machine learning, data mining, and statistics. They offer a clear and intuitive method for decision-making by modeling the relationships between different variables based on data.

1.1.1 What is a Decision Tree

A decision tree resembles a flowchart and serves to make decisions or predictions. It comprises nodes that depict decisions or attribute tests, branches that illustrate the consequences of these decisions, and leaf nodes that signify ultimate outcomes or predictions. Internal nodes are linked to attribute tests, branches reflect test results, and leaf nodes denote class labels or continuous values.

1.1.2 Structure of a Decision Tree

A **decision tree** is a tree-like model used for decision making, where each node represents a test or decision on an attribute, each branch represents the outcome of that test, and each leaf node represents a class label (for classification) or a continuous value (for regression).

Here's a detailed breakdown of its structure:

1. Root Node

- **Definition**: The topmost node in the tree, representing the entire dataset or problem. It is the starting point of the decision-making process.
- **Function**: Splits the data based on the best attribute (feature) that leads to the most significant separation.

2. Internal Nodes (Decision Nodes)

- **Definition**: Nodes that represent decisions or tests on features. They are intermediate nodes that guide the splitting process.
- **Function**: Each internal node splits the dataset into subsets based on the outcome of a feature test.

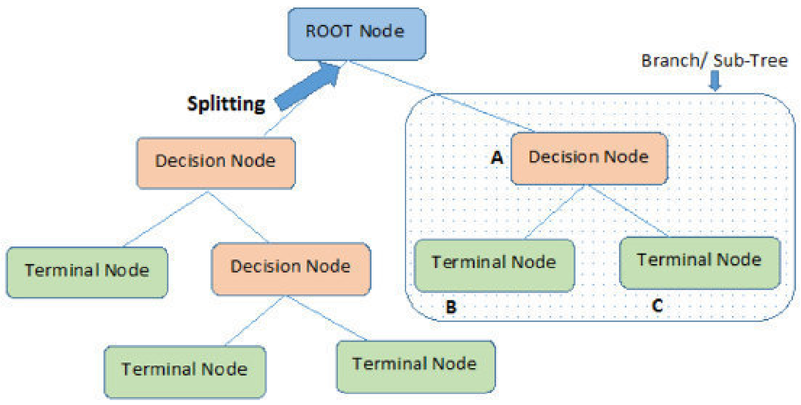
- **Components**:
    - **Feature Test**: A condition on a feature that determines how the data is split (e.g., if a feature value is greater than a threshold).

3. Branches (Edges)

- **Definition**: Arrows connecting nodes, representing the outcomes of feature tests.
- **Function**: Direct the flow from one node to another based on the results of the feature test (e.g., true/false, yes/no).

4. Leaf Nodes (Terminal Nodes)

- **Definition**: Nodes that do not split further. They represent the final output or decision.
- **Function**: Provide the outcome for the path taken through the tree. In classification, they assign a class label; in regression, they provide a value.
- **Components**:
    - **Class Label**: In classification, the label assigned (e.g., "fit" or "unfit").
    - **Value**: In regression, the continuous output value.



1.1.3 Example of a Decision Tree

An example of a decision tree can be explained using below binary tree. Let's say you want to predict whether a person is fit given their information like age, eating habit, and physical activity, etc. The decision nodes here are questions like 'What's the age?', 'Does he exercise?', 'Does he eat a lot of pizzas'? And the leaves, which are outcomes like either 'fit', or 'unfit'. In this case this was a binary classification problem (a yes no type problem).

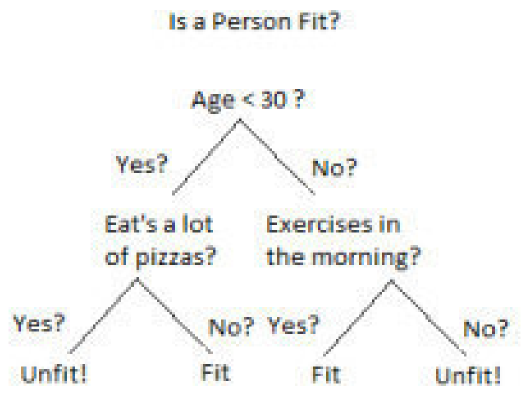


Figure: Decision tree

There are two main types of Decision Trees:

1. ***Classification trees*** *(Yes/No types)*

   What we've seen above is an example of a classification tree, where the outcome was a categorical variable such as 'fit' or 'unfit'.

2. ***Regression trees*** *(Continuous data types)*

   Here, the decision or outcome variable is continuous, such as a number like 123.

Now that we understand what a Decision Tree is, let's explore how it functions internally. There are numerous algorithms available for constructing Decision Trees, but one of the most effective is the ID3 Algorithm, which stands for Iterative Dichotomiser 3. Before delving into the ID3 algorithm, we'll review a few key definitions.

How It Works

1. **Splitting**: Starting from the root, the tree splits based on feature tests. Each split aims to partition the data into more homogenous groups regarding the target variable.
2. **Decision Path**: As data moves from the root to the leaves, each decision narrows down the possibilities.
3. **Outcome**: Once a leaf is reached, the decision (class or value) is assigned based on the majority class (in classification) or average value (in regression) of the data points in that leaf.

Key Characteristics

- **Hierarchical Structure**: Decisions are made in a top-down approach.
- **Non-linear**: Decision trees can model non-linear relationships.
- **Interpretability**: They are easy to understand and interpret as the decision-making process mimics human reasoning.

Additional Concepts

- **Pruning**: Removing parts of the tree that do not provide additional power to improve generalization and prevent over fitting.

- **Tree Depth**: The maximum distance between the root and a leaf node. Deeper trees can model more complex patterns but may over fit.

1.1.4 Advantages and Disadvantages of Decision Tree

Advantages:

- **Easy to Understand**: Simple to visualize and interpret.
- **Versatile**: Can handle both numerical and categorical data.
- **Little Data Preparation**: Requires little preprocessing (e.g., normalization).

Disadvantages:

- **Over fitting**: Can easily over fit, especially if the tree is very deep.
- **Instability**: Small changes in the data can result in a completely different tree structure.

1.2 Defining the Classification Problem

Defining a classification problem specifically for a decision tree involves focusing on how the problem will be framed to leverage the strengths of decision trees.

1.2.1 Steps for Defining a Classification Problem:

1. Problem Definition

- **Objective**: Clearly state the goal of the classification. For example, predicting customer churn, diagnosing a disease, or classifying emails as spam or not spam.

- **Classes**: Identify the distinct categories or classes. Decision trees handle both binary (e.g., yes/no) and multi-class (e.g., species classification) problems well.

2. Data Collection

- **Data Sources**: Collect data from relevant sources such as databases, surveys, sensors, or other input mechanisms.
- **Attributes/Features**: Determine which features will be used for prediction. Decision trees can handle both numerical and categorical data.

3. Data Preparation

- **Cleaning**: Handle missing values (e.g., imputation), remove duplicates, and address outliers.
- **Transformation**: Encode categorical variables (e.g., one-hot encoding), normalize or standardize numerical features if necessary.
- **Splitting**: Divide the data into training and testing sets, typically using a 70-30 or 80-20 split.

4. Feature Selection

- **Relevance**: Use domain knowledge and statistical methods (e.g., correlation analysis) to select relevant features.
- **Reduction**: Eliminate redundant or irrelevant features. Decision trees perform implicit feature selection, but preprocessing can still be beneficial.

5. Model Selection

- **Algorithm**: Choose a decision tree algorithm (e.g., CART, ID3, C4.5).

- **Hyper parameters**: Determine hyper parameters such as tree depth, minimum samples per leaf, and split criteria (e.g., Gini impurity, information gain).

6. Training the Model

- **Parameter Tuning**: Use techniques like grid search or random search to optimize hyper parameters.
- **Cross-Validation**: Use k-fold cross-validation to ensure the model's generalizability.

7. Evaluation

- **Metrics**: Choose appropriate evaluation metrics like accuracy, precision, recall, F1 score, and ROC-AUC.
- **Confusion Matrix**: Analyze the confusion matrix to understand the types of errors made by the model.
- **Over fitting Check**: Ensure the model is not over fitting by comparing performance on training and testing sets.

8. Deployment and Monitoring

- **Implementation**: Deploy the trained decision tree model into a production environment.
- **Monitoring**: Continuously monitor model performance and update the model with new data as necessary.

1.2.2 Example: Decision Tree for Predicting Customer Churn

1. Problem Definition

- **Objective**: Predict whether a customer will churn (leave the service) based on their usage patterns and demographics.
- **Classes**: Churn (yes), No Churn (no).

2. Data Collection

- **Data Sources**: Customer transaction records, demographic information, service usage logs.
- **Attributes/Features**: Age, tenure, monthly charges, total charges, usage frequency, customer service calls, etc.

3. Data Preparation

- **Cleaning**: Handle missing values by imputation, remove duplicate records.
- **Transformation**: Encode categorical features (e.g., gender, contract type), normalize numerical features.
- **Splitting**: Split data into 70% training and 30% testing sets.

4. Feature Selection

- **Relevance**: Select features based on domain knowledge and statistical analysis (e.g., correlation with churn).
- **Reduction**: Remove highly correlated or irrelevant features to simplify the model.

5. Model Selection

- **Algorithm**: Choose the CART (Classification and Regression Trees) algorithm.
- **Hyper parameters**: Set maximum tree depth to avoid overfitting, minimum samples per leaf, and choose Gini impurity as the split criterion.

6. Training the Model

- **Parameter Tuning**: Use grid search to find the optimal tree depth and minimum samples per leaf.

- **Cross-Validation**: Perform 5-fold cross-validation to validate the model's performance.

7. Evaluation

- **Metrics**: Calculate accuracy, precision, recall, F1 score, and ROC-AUC.
- **Confusion Matrix**: Analyze the confusion matrix to identify false positives and false negatives.
- **Over fitting Check**: Compare performance on training and testing sets to detect over fitting.

8. Deployment and Monitoring

- **Implementation**: Integrate the decision tree model into the customer relationship management (CRM) system.
- **Monitoring**: Regularly monitor the model's performance on new data and retrain the model periodically to maintain accuracy.

By following these steps, we can effectively define and address a classification problem using a decision tree, ensuring the model is well-prepared to make accurate predictions.

SAQ

1. Define a Classification Problem taking one real example.

1.3 Tree Construction Principle

The construction of a decision tree involves several key principles and steps.

1.3.1Feature Selection:

At each node of the tree, the algorithm selects the feature that best splits the data according to some criterion. Common criteria include:

- **Gini Impurity**: Used in the CART (Classification and Regression Tree) algorithm. It measures the impurity of a node. A lower Gini impurity indicates a better split.
- **Information Gain**: Used in the ID3 (Iterative Dichotomiser 3) and C4.5 algorithms. It measures the reduction in entropy after the dataset is split on an attribute.
- **Chi-square**: Used in CHAID (Chi-squared Automatic Interaction Detector) to test the statistical significance of the splits.
- **Reduction in Variance**: Often used in regression trees to measure the reduction in variance of the target variable.

1.3.2Tree Building Process:

The decision tree is built using a recursive process:

- **Start with the entire dataset**: Choose the best feature to split the data.
- **Split the dataset**: Divide the data into subsets based on the chosen feature. Each subset corresponds to a branch of the tree.
- **Repeat recursively**: For each subset, repeat the process of selecting the best feature and splitting the data. This continues until one of the stopping criteria is met.

1.3.3Stopping Criteria:

The recursive splitting process stops when one of the following conditions is met:

- **Maximum Tree Depth**: The tree reaches a specified maximum depth.
- **Minimum Node Size**: A node is not split if it contains fewer than a specified number of instances.
- **Purity of Node**: A node is considered pure if all instances belong to a single class.
- **Minimum Information Gain**: The information gain of a split is below a specified threshold.

1.3.4Pruning:

To avoid overfitting, pruning techniques are applied to the tree:

- **Pre-pruning (Early Stopping)**: The tree-building process is stopped early based on certain criteria (e.g., maximum depth, minimum node size).
- **Post-pruning**: After the tree is fully grown, some branches are removed. Techniques include:
  - **Cost-Complexity Pruning (CCP)**: Balances the trade-off between the tree's complexity and its performance on the training data.
  - **Reduced Error Pruning**: Removes branches only if it improves the tree's performance on a validation dataset.

1.3.5Handling Missing Values:

Decision trees can handle missing values in different ways:

- **Surrogate Splits**: Use alternate features to split the data when the primary feature has missing values.

- **Missing Value Imputation**: Fill in missing values using statistical methods like mean, median, or most frequent value.

1.3.6 Handling Categorical Features:

Categorical features can be handled in several ways:

- **One-Hot Encoding**: Convert categorical features into binary features.
- **Label Encoding**: Assign a unique integer to each category.
- **Binary Splits**: Create splits based on specific categories.

1.3.7 Example of Decision Tree Construction:

Here is a simplified example of constructing a decision tree for a classification task:

1. **Select the best feature**: Using criteria like Gini impurity or information gain.
2. **Split the data**: Create branches based on the selected feature's values.
3. **Repeat for each branch**: Continue the process recursively until stopping criteria are met.
4. **Prune the tree**: Remove branches that do not improve performance.

SAQ

1. What do you mean by feature selection?
2. What is over fitting? How pruning technique helps to reduce over fitting?

1.4 Best Split

Node splitting, or simply splitting, divides a node into multiple sub-nodes to create relatively pure nodes. This is done by finding the best split for a node and can be done in multiple ways. The "best" split in a decision tree refers to the decision rule at a node that best separates the data into classes. The quality of a split is typically evaluated using a criterion, which can vary depending on the type of task (classification or regression).

The ways of splitting a node can be broadly divided into two categories based on the type of target variable:

1. **Continuous Target Variable:** Reduction in Variance
2. **Categorical Target Variable:** Gini Impurity, Information Gain, and Chi-Square

1.4.1 Reduction in Variance in Decision Tree

Reduction in Variance is a method for splitting the node used when the target variable is continuous, i.e., regression problems. It is called so because it uses variance as a measure for deciding the feature on which a node is split into child nodes.

$$Variance = \frac{\sum (X - \mu)^2}{N}$$

Variance is used for calculating the homogeneity of a node. If a node is entirely homogeneous, then the variance is zero.

Here are the steps to split a decision tree using the reduction in variance method:

1. For each split, individually calculate the variance of each child node

2. Calculate the variance of each split as the weighted average variance of child nodes
3. Select the split with the lowest variance
4. Perform steps 1-3 until completely homogeneous nodes are achieved

1.4.2 Information Gain in Decision Tree

Now, what if we have a categorical target variable? For categorical variables, a reduction in variation won't quite cut it. Well, the answer to that is Information Gain. The Information Gain method is used for splitting the nodes when the target variable is categorical. It works on the concept of entropy and is given by:

$$Information\,Gain = 1 - Entropy$$

Entropy is used for calculating the purity of a node. **The lower the value of entropy, the higher the purity of the node.** The entropy of a homogeneous node is zero. Since we subtract entropy from 1, the Information Gain is higher for the purer nodes with a maximum value of 1. Now, let's take a look at the formula for calculating the entropy:

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Steps to split a decision tree using Information Gain:

1. For each split, individually calculate the entropy of each child node
2. Calculate the entropy of each split as the weighted average entropy of child nodes

3. Select the split with the lowest entropy or highest information gain
4. Until you achieve homogeneous nodes, repeat steps 1-3

1.4.3 Gini Impurity in Decision Tree

Gini Impurity is a method for splitting the nodes when the target variable is categorical. It is the most popular and easiest way to split a decision tree. The Gini Impurity value is:

$$Gini\ Impurity = 1 - Gini$$

Gini is the probability of correctly labeling a randomly chosen element if it is randomly labeled according to the distribution of labels in the node. The formula for Gini is:

$$Gini = \sum_{i=1}^{n} p_i^2$$

And Gini Impurity is:

$$Gini\ Impurity = 1 - \sum_{i=1}^{n} p_i^2$$

The lower the Gini Impurity, the higher the homogeneity of the node. **The Gini Impurity of a pure node is zero.** Now, you might be thinking we already know about Information Gain then, why do we need Gini Impurity?

Gini Impurity is preferred to Information Gain because it does not contain logarithms which are computationally intensive.

Here are the steps to split a decision tree using Gini Impurity:

1. Similar to what we did in information gain. For each split, individually calculate the Gini Impurity of each child node
2. Calculate the Gini Impurity of each split as the weighted average Gini Impurity of child nodes
3. Select the split with the lowest value of Gini Impurity
4. Until you achieve homogeneous nodes, repeat steps 1-3

1.4.4 Chi-Square in Decision Tree

Chi-square is another method of splitting nodes in a decision tree for datasets having categorical target values. It is used to make two or more splits in a node. It works on the statistical significance of differences between the parent node and child nodes.

The Chi-Square value is:

$$Chi\text{-}Square = \sqrt{\frac{(Actual\ \ Expected)^2}{Expected}}$$

Here, the *Expected* is the expected value for a class in a child node based on the distribution of classes in the parent node, and the *Actual* is the actual value for a class in a child node.

The above formula gives us the value of Chi-Square for a class. Take the sum of Chi-Square values for all the classes in a node to calculate the Chi-Square for that node. The higher the value, the higher will be the differences between parent and child nodes, i.e., the higher will be the homogeneity.

Here are the steps to split a decision tree using Chi-Square:

1. For each split, individually calculate the Chi-Square value of each child node by taking the sum of Chi-Square values for each class in a node
2. Calculate the Chi-Square value of each split as the sum of Chi-Square values for all the child nodes
3. Select the split with a higher Chi-Square value
4. Until you achieve homogeneous nodes, repeat steps 1-3

Decision trees are an important tool in machine learning for solving classification and regression problems. However, creating an effective decision tree requires choosing the right features and splitting the data in a way that maximizes information gain.

1.5 Splitting indices

Splitting indices in a decision tree refer to the indices of the dataset that belong to the left or right child node after a split at a particular node. To determine these indices, the decision tree algorithm evaluates a split condition (e.g., feature <= threshold) for each data point at the current node. Here's how the process generally works:

1. **Identify the Split Condition:** The decision tree algorithm selects the best feature and threshold that maximizes a certain criterion (e.g., Gini impurity, entropy) for the split.
2. **Evaluate the Split Condition:** For each data point at the current node, evaluate whether it satisfies the split condition.
3. **Assign Data Points to Child Nodes:** Based on the evaluation:
   - If a data point satisfies the condition (e.g., feature <= threshold), it is assigned to the left child node.

- Otherwise, it is assigned to the right child node.

1.6 Splitting criteria

The splitting criteria of a decision tree determine how the nodes of the tree are split into child nodes based on the features of the dataset. The primary goal of the splitting criteria is to maximize the separation of the data according to the target variable.

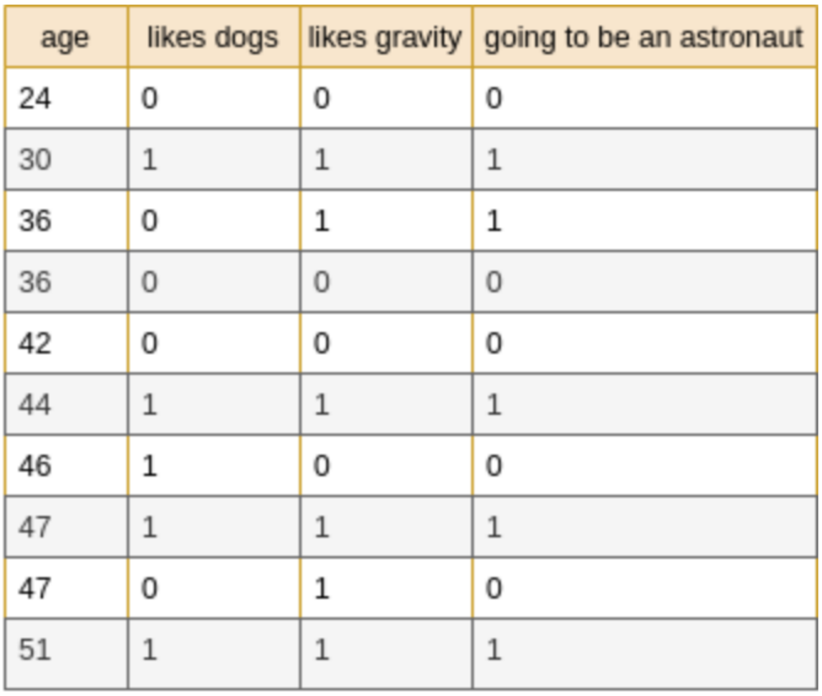
How to Determine the Best Split in Decision Tree:

To determine the best split in a decision tree, select the split that maximizes information gain or minimizes impurity. The steps are:

1. **Calculate Impurity Measure**:
   - Compute an impurity measure (e.g., Gini impurity or entropy) for each potential split based on the target variable's values in the resulting subsets.
2. **Calculate Information Gain**:
   - For each split, calculate the information gain, which is the reduction in impurity achieved by splitting the data.
3. **Select Split with Maximum Information Gain**:
   - Choose the split that maximizes information gain. This split effectively separates the data into subsets that are more homogeneous with respect to the target variable.
4. **Repeat for Each Attribute**
   - Repeat the process for all available attributes, selecting the split with the highest information gain across attributes

Decision tree for classification: an example:

Traditionally decision trees are drawn manually, but they can be learned using Machine Learning. They can be used for both regression and classification problems. Here we will discuss the classification problems. Let’s consider the following example data:



age	likes dogs	likes gravity	going to be an astronaut
24	0	0	0
30	1	1	1
36	0	1	1
36	0	0	0
42	0	0	0
44	1	1	1
46	1	0	0
47	1	1	1
47	0	1	0
51	1	1	1

Figure: Example Dataset

Using this simplified example, we will predict whether a person is going to be an astronaut based on their age, their preference for dogs, and their opinion on gravity. Let us examine the resulting decision tree for our example data.

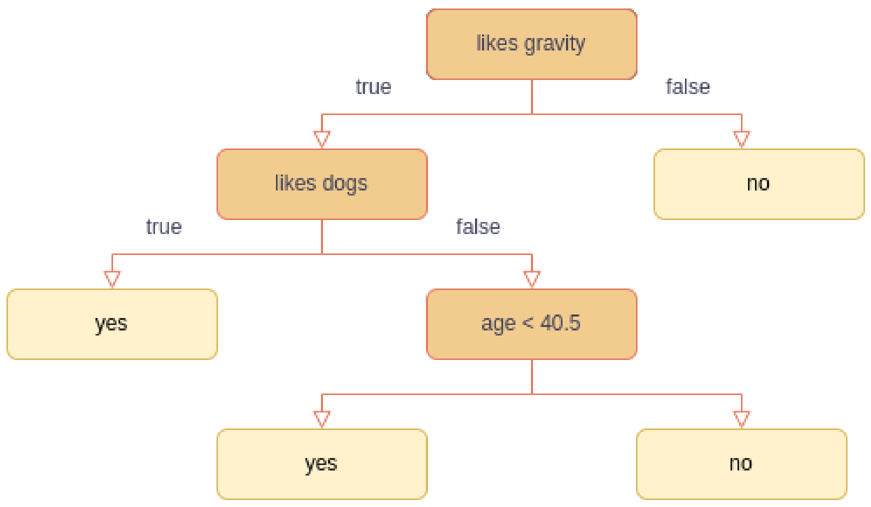


Figure: Final decision tree for example data

We can follow different paths to arrive at a decision. For instance, we can determine that someone who dislikes gravity is not going to be an astronaut, regardless of other characteristics. Conversely, we can see that someone who likes gravity and likes dogs is likely to become an astronaut, irrespective of their age.

Create a Decision Tree

The most crucial step in creating a decision tree is splitting the data. We need to find a method to divide the data set (D) into two subsets (D_1 and D_2). Various criteria can be used to determine the best split. Here, we will focus on one such criterion: Gini Impurity, which is used for categorical target variables.

Gini Impurity

The Gini Impurity for a data set *D* is calculated as follows:

$$\text{Gini Impurity}(D) = \frac{n_1}{n} \cdot \text{Gini}(D_1) + \frac{n_2}{n} \cdot \text{Gini}(D_2)$$

with n = $n_1 + n_2$ the size of the data set (D) and

$$\text{Gini}(D_j) = 1 - \sum_{j=1}^{c} p_j^2$$

with D_1 and D_2subsets of *D*, p_j the probability of samples belonging to class *j* at a given node, and *c* the number of classes. The lower the Gini Impurity, the higher is the homogeneity of the node. The Gini Impurity of a pure node is zero. To split a decision tree using Gini Impurity, the following steps need to be performed.

1. For each possible split, calculate the Gini Impurity of each child node
2. Calculate the Gini Impurity of each split as the weighted average Gini Impurity of child nodes
3. Select the split with the lowest value of Gini Impurity

Repeat steps 1–3 until no further split is possible.

To understand this better, let’s have a look at an example.

First Example: Decision Tree with two binary features

Before creating the decision tree for our entire dataset, we will first consider a subset, that only considers two features: ‘likes gravity’ and ‘likes dogs’.
First, we need to determine which feature will be the root node. To do this, we predict the target using only one feature at a time and select the feature with the lowest Gini Impurity as the root node. In our case, we construct two shallow trees, each with just a root node

and two leaves. In the first tree, we use 'likes gravity' as the root node, and in the second tree, we use 'likes dogs'. We then calculate the Gini Impurity for both scenarios. The trees look like this:

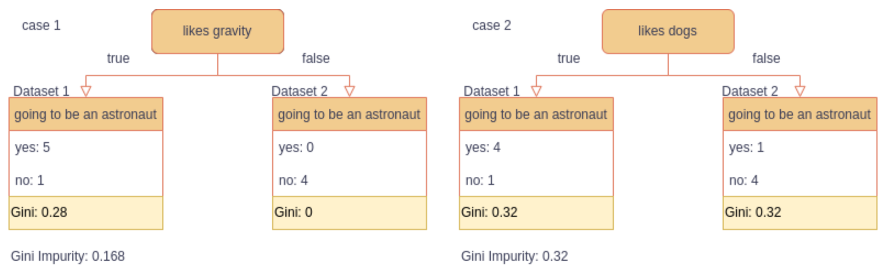


Figure: Shallow tree

The Gini Impurity for these trees are calculated as follows:

Case 1:

Dataset 1:

$$\text{Gini}(D_1) = 1 - \left(\frac{5}{5+1}\right)^2 - \left(\frac{1}{5+1}\right)^2 = 0.28$$

Dataset 2:

$$\text{Gini}(D_2) = 1 - \left(\frac{0}{0+4}\right)^2 - \left(\frac{4}{0+4}\right)^2 = 0$$

The Gini Impurity is the weighted mean of both:

$$\text{Gini Impurity} = \frac{6}{10} \cdot 0.28 + \frac{4}{10} \cdot 0 = 0.168$$

Case 2:

Dataset 1:

$$\text{Gini}(D_1) = 1 - \left(\frac{4}{1+4}\right)^2 - \left(\frac{1}{1+4}\right)^2 = 0.32$$

Dataset 2:

$$\text{Gini}(D_2) = 1 - \left(\frac{1}{1+4}\right)^2 - \left(\frac{4}{1+4}\right)^2 = 0.32$$

The Gini Impurity is the weighted mean of both:

$$\text{Gini Impurity} = \frac{5}{10} \cdot 0.32 + \frac{5}{10} \cdot 0.32 = 0.32$$

That is, the first case has lower Gini Impurity and is the chosen split. In this simple example, only one feature remains, and we can build the final decision tree.

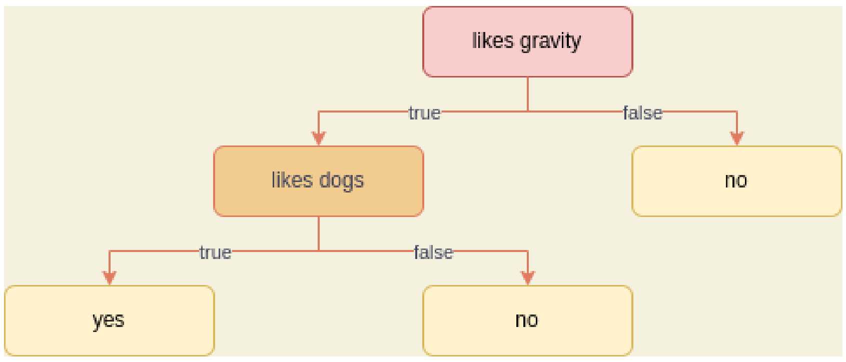


Figure: Decision tree

Stop to Consider

Above example is taken from www.towardsdatascience.com

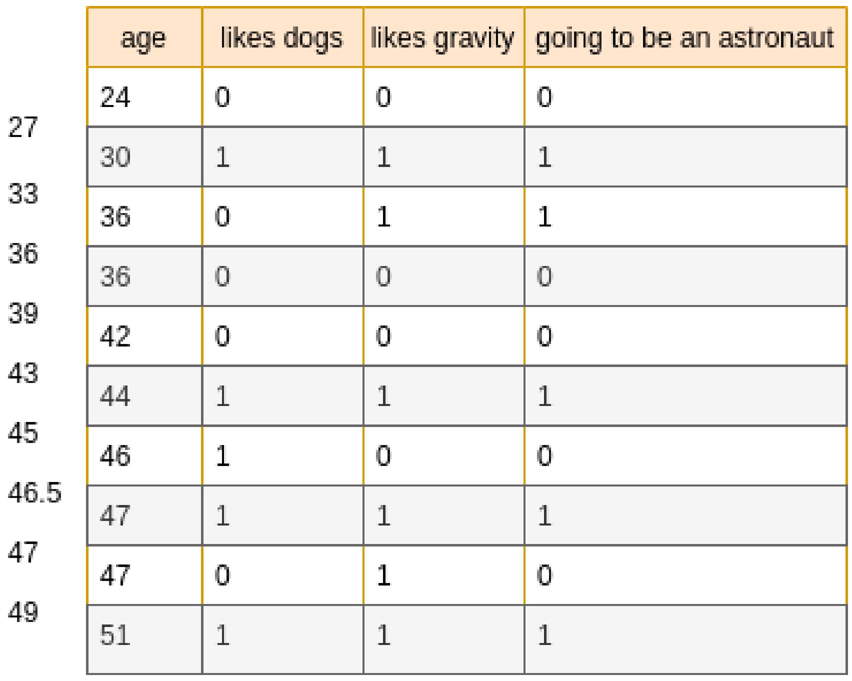
Students can follow different sites for more examples.

Second Example: Add a numerical Variable

So far, we have only considered the categorical variables in our dataset. Now, we will include the numerical variable 'age'. The splitting criterion remains the same. We already know the Gini Impurities for 'likes gravity' and 'likes dogs'. Calculating the Gini Impurity for a numerical variable is similar, but it involves more steps. The following steps need to be completed:

1. Sort the data frame by the numerical variable (‘age’)
2. Calculate the mean of neighbouring values
3. Calculate the Gini Impurity for all splits for each of these means

Here is our data, now sorted by age, with the mean of neighboring values provided on the left-hand side.



27
33
36
39
43
45
46.5
47
49

age	likes dogs	likes gravity	going to be an astronaut
24	0	0	0
30	1	1	1
36	0	1	1
36	0	0	0
42	0	0	0
44	1	1	1
46	1	0	0
47	1	1	1
47	0	1	0
51	1	1	1

Figure: The data set sorted by age. The left hand side shows the mean of neighbouring values for age.

We then have the following possible splits.

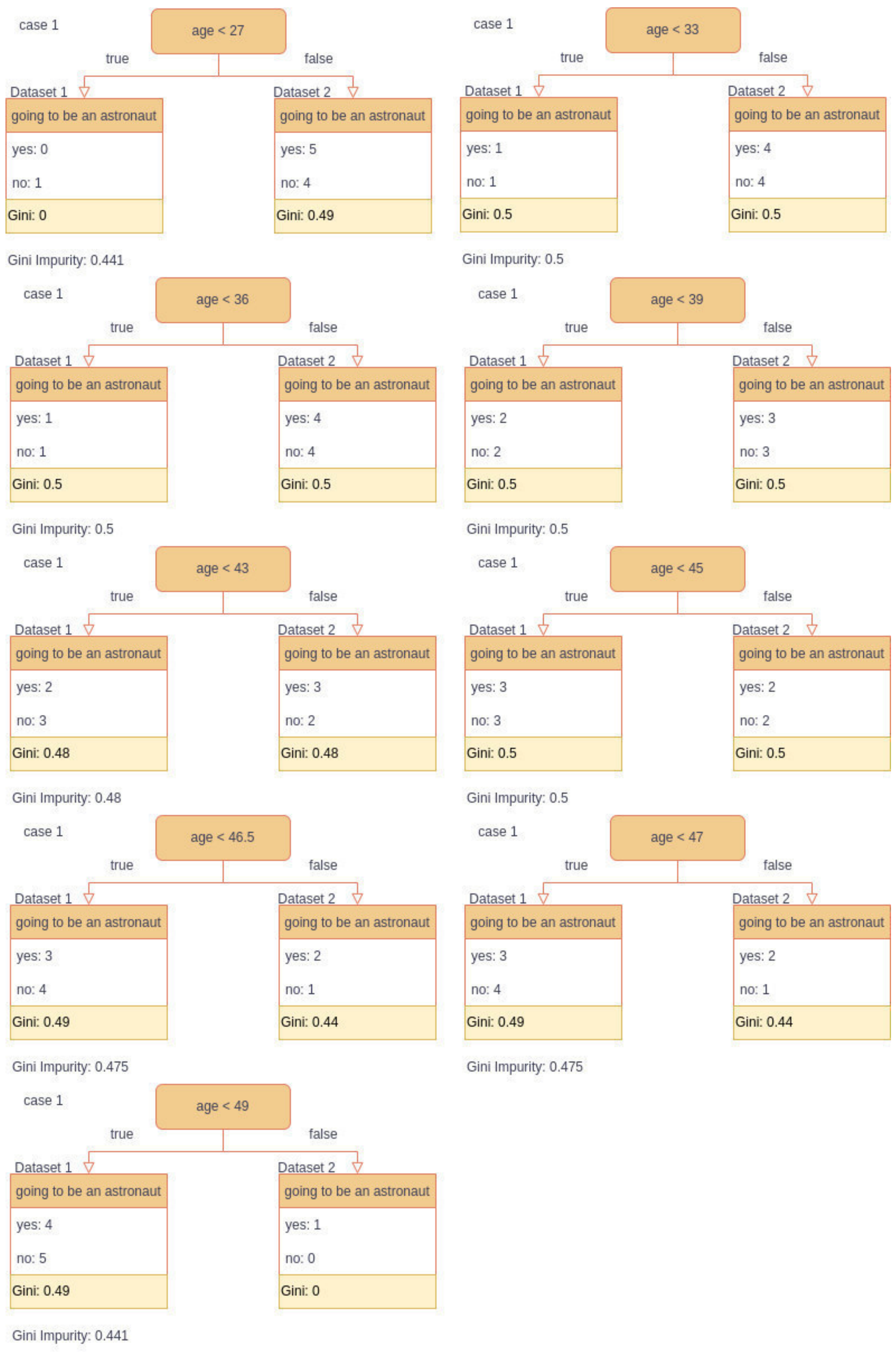
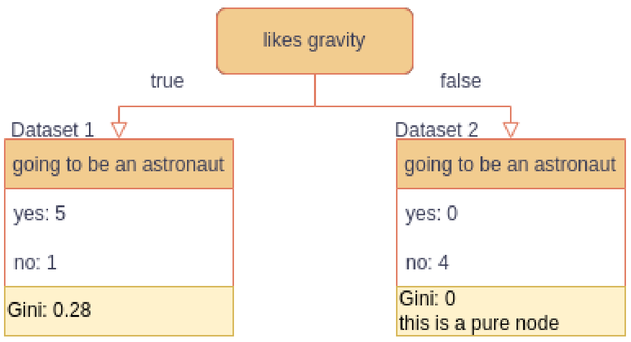


Figure: Possible splits for age and their Gini Imputity.

We can see that the Gini Impurity of all possible ‘age’ splits is higher than the one for ‘likes gravity’ and ‘likes dogs’. The lowest Gini Impurity occurs when using 'likes gravity', making it our root node and the first split.



age	likes dogs	going to be an astronaut
30	1	1
36	0	1
44	1	1
47	1	1
47	0	0
51	1	1

Figure: The first split of the tree. ‘likes gravity’ is the root node.

The subset Dataset 2 is already pure, meaning this node is a leaf and no further splitting is required. However, the left-hand branch, Dataset 1, is not pure and can be split further. To do this, we follow the same process as before: calculating the Gini Impurity for each feature, namely 'likes dogs' and 'age'.

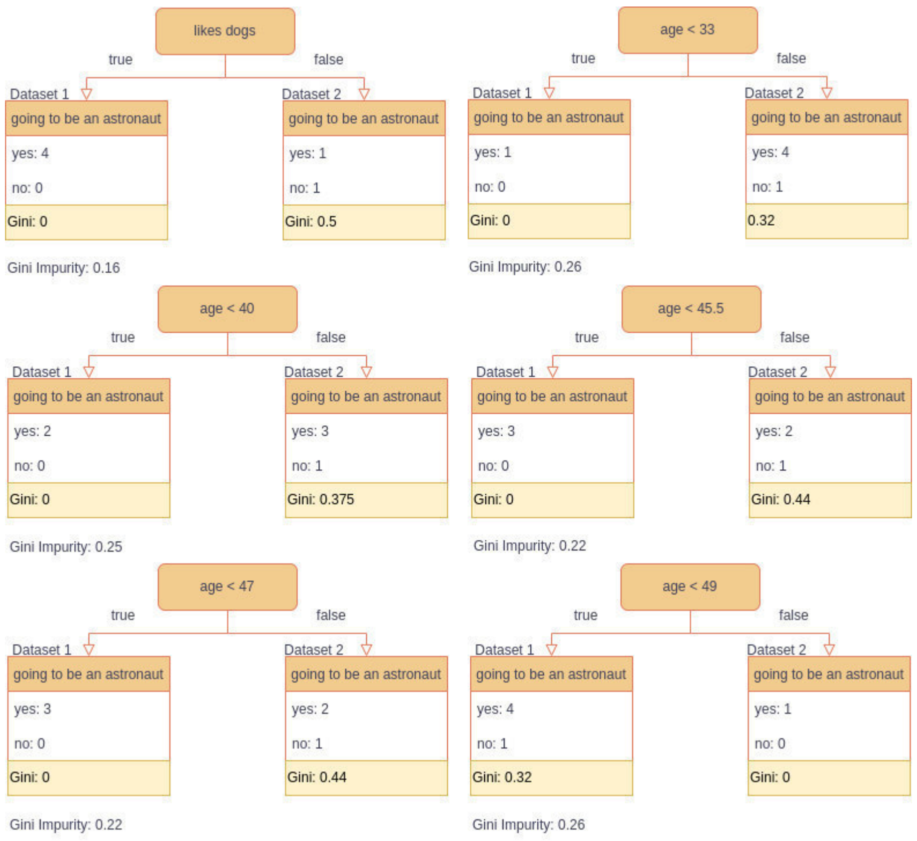


Figure: Possible splits for Dataset 2.

The split "likes dogs" results in the lowest Gini Impurity. With this information, we can now construct our final tree.

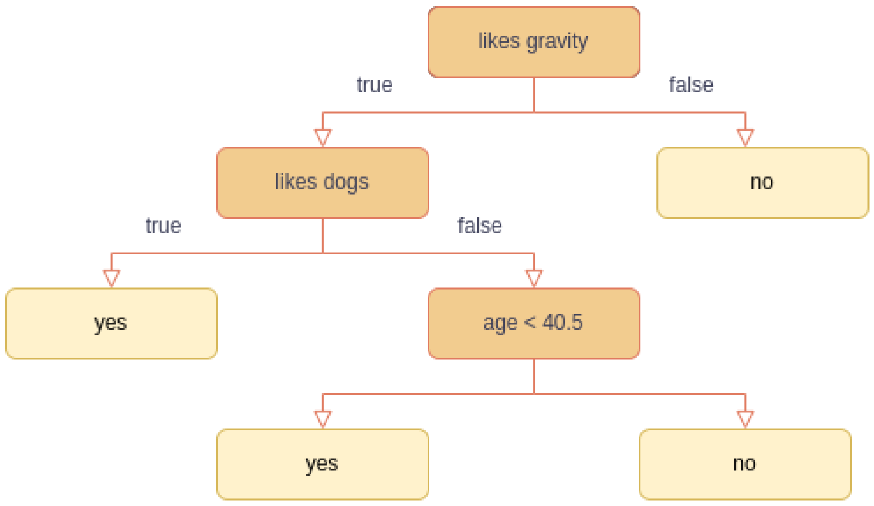


Figure: Final Decision Tree.

SAQ

1. Draw a regression tree taking one example and discuss the steps followed.

1.7 Summing Up

- A **decision tree** is a tree-like model used for decision making.
- Defining a classification problem for a decision tree involves focusing on how the problem will be framed.
- The construction of a decision tree involves several key principles and steps.
- The quality of a split is typically evaluated using some criterion.

1.8 Possible Questions

1. What is Decision Tree?
2. Discuss the advantages and disadvantages of Decision tree.
3. What is best split in a Decision tree?
4. What are different criterions to be consider for splitting a node.
5. Define the various terms related with structure of a Decision tree.
6. Discuss the steps for defining a Classification Problem.
7. Discuss the steps for Decision tree construction.
8. Discuss the steps of classification tree construction with an example.

1.9 References and Suggested Readings

2. www.geeksforgeeks.org

2. www.javatpoint.com

3. www.analyticsvidhya.com

4. www.towardsdatascience.com

---×---

UNIT: 2
DECISION TREE GENERATION ALGORITHM

Unit Structure:

2.1 Introduction

- It is a graphical representation for getting solutions to a problem/decision based on given conditions
- Similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure that's why it is called a decision tree.

- Based on questions, it answers YES/No and further the tree splits into subtrees.
- The Decision Node and Leaf Node are the two nodes in a decision tree. To make any decision, decision nodes are used and have multiple branches, where as the output of those decisions is leaf nodes that do not contain any further branches.
- Based on features of the given dataset, tests or decisions are performed.
- For both classification and Regression problems, a decision tree is used. But mostly classification problem is preferred. It is a tree structure where each leaf node represents the outcome, branches represent the decision rules and internal nodes represent the features of a dataset**.**

2.2 Objectives

After going through this unit student will be able to learn

1. The concept of Decision tree
2. Attribute Selection Measures
3. Why use Decision Tree?
4. Different types of Decision tree
5. Will be able to real-life problems in the form of decision tree.

2.3 Basic Concept and Terms of Decision Tree

- **Root Node:** It is the highest node representing the feature from which the tree branches.
- **Internal Nodes (Decision Nodes)**: Internal nodes in the tree are the nodes where the test is performed and branches are available to this node that goes to another node.

- **Leaf Nodes (Terminal Nodes)**: The leaf node of a tree represents classes and from the leaf node, there are no further branches on the leaf nodes.
- **Branches (Edges)**: Branches represent the link between two nodes and show the response while the test is performed on the internal node.
- **Splitting**: Based on a decision the process of splitting a node into two or more sub-nodes.
- **Parent Node**: The node from where branches come is the Parent node.
- **Child Node**: A parent node splits as a result child node creates.
- **Decision Criterion**: It is a condition based on which data should be split on a decision node. Features are compared against a threshold value.
- **Pruning**: Pruning involves removing nodes and branches thus improving generalization and preventing over fitting.

2.4 Why use Decision Trees?

While creating a machine learning model, there are many algorithms available but we have to select the appropriate one based on the problem and dataset on hand.

Below are the two reasons for using the Decision tree:

- It is easy to understand Decision Trees because they imitate thinking just like a human.
- The tree-like structure makes us understand the logic very easily.

2.5 Attribute Selection Measures

How to select the best node as a root node for a tree as well as for a subtree is an issue. So we have a technique called Attribute selection measure or ASM. Two techniques are available

- Information Gain
- Gini Index

1. Information Gain:

- After segmentation of a dataset based on an attribute, a change in entropy occurs and the measurement of changes is nothing but information gain.
- It calculates how much information a feature provides us about a class.
- We split the node and build the decision tree based on the value of information gain.
- A node/attribute having the highest information gain is split first. It can be calculated as:

1. Information Gain= Entropy(S)- [(Weighted Avg) *Entropy (each feature)

Entropy: It Measures the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

- S= Total number of samples
- P(yes)= probability of yes
- P(no)= probability of no

2. Gini Index:

- We use Gini Index in Decision Tree multiple times while splitting up the nodes. It is a measure of impurity or purity.
- A low Gini index attribute should be preferred as compared to the high Gini index.
- The Gini index can be calculated using the below formula:

Gini Index= $1- \sum_j P_j^2$

STOP TO CONSIDER

To accurately describe some sample, information is needed which is nothing but Entropy. If all the elements are similar then Entropy is said to be 0, otherwise, if the sample is equally divided then entropy is a maximum of 1.

Gini index values fall between 0 and 1. When samples are perfectly homogeneous, the Gini index value is 0 and when maximum inequality exists among elements, the Gini index value is 1.

CHECK YOUR PROGRESS

1. What are the disadvantages of Information Gain?
2. List down the problem domains in which Decision Trees are most suitable.
3. List down some popular algorithms used for deriving Decision Trees and their attribute selection measures.
4. List down the different types of nodes in Decision Trees.
5. What do you understand about Information Gain?

2.6 Types of Decision Tree

In this chapter we mainly going to study three types of Decision tree:

1. CART

2. ID3

3. C4.5

2.6.1 CART (Classification and Regression Tree)

CART is a decision-based algorithm used both for classification and Regression problems. Using binary splits, it recursively partitioned the training data sets. It handles both categorical and continuous features that's why it is a very powerful and popular algorithm. Each terminal node has two child nodes which differ from another tree-based method that allows multiple child nodes. The root node initially contains all the training data and until a stopping criterion is reached, nodes are recursively split into smaller subsets.

A node that has maximum information gain or Gini impurity will be selected by the algorithm as a Root node. The Gini Index measures how well it will split the data.

The process continues recursively, with each node in the tree splitting the data into two smaller subsets until a stopping criterion is met. Until a stopping criterion is met. this process recursively splits the data into smaller data subsets. The stopping criterion may be when the tree reaches maximum depth or a minimum number of instances in each leaf node, or other criteria.

Predictions can be done once the tree is built, by traversing the tree from the root node to a leaf node. The average of the target values in the leaf node is used for the regression problem. For classification problems, the majority of classes in the leaf node are used for classification problem.

2.6.1.1 Steps of CART algorithm

1. Calculate the Gini impurity for all input variables. The minimum Gini impurity is chosen as a split point.
2. Based on the chosen split point, data is divided into two subsets, and a new node is selected for each subset.
3. Repeat 2 and 3 until a criterion is met to stop it. Stopping criteria may be a minimum number of data points in a leaf node or a maximum tree depth or when the impurity is minimum.
4. After that the tree obtained is the decision tree.

2.6.1.2 Advantages and disadvantages of CART algorithm

Advantages of the CART algorithm

1.It is simple and easy to understand, interpret and visualize.

2. Both numerical and categorical data are handled by CART

3.By imputing them with surrogate splits, missing values are handled. The surrogate splits the data in exactly the same way as the primary split.

4.Performs automatic feature selection.

5. Interaction effects between features are easily possible.

Disadvantages of CART algorithm:

1. High instability of the trees.
2. Small changes in the data may lead to very different trees.
3. Because of this reason prediction error of the tree is usually high.

Problem 1. Consider the following data set. There are 14 instances of golf playing instances based on outlook, temperature, humidity, and wind factors. Draw a decision tree using the CART algorithm.

Day	Outlook	Temp	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Strong	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Solution:

1. First we need to calculate Gini Index

   a) Gini Index for Outlook

Outlook	**Yes**	**No**	**Number of Instance**
Sunny	2	3	5
Overcast	4	0	4
Rain	3	2	5

Gini(Outlook=Sunny) =1- $(2/5)^2 - (3/5)^2$ =1 - 0.16-0.36=0.48

Gini (Outlook=Overcast) =1- $(4/4)^2 - (0/4)^2$ =0

Gini(Outlook=Rain) =1 – $(3/5)^2 - (2/5)^2$=1 – 0.36 – 0.16=0.48

Now will calculate total Gini Index for outlook feature

Gini(Outlook)= (5/14) *0.48+(4/14) *0+(5/14) *0.48=0.171+0+0.171=0.342

b) Now we will calculate Gini Index for Temperature:

Temperature has 3 different values Hot, Cool and Mild.

Temperature	**Yes**	**No**	**Number of Instances**
Hot	2	2	4
Cool	3	1	4
Mild	4	2	6

Gini(Temperature=Hot) =1-$(2/4)^2$-$(2/4)^2$=0.5

Gini(Temp=Cool) =1- $(3/4)^2$-$(1/4)^2$=1 – 0.5625 -0.0625=0.375

Gini(Temp=Mild) =1 – $(4/6)^2$- $(2/6)^2$ =1-0.444-0.111=0.445

Now will calculate total Gini Index for Temperature feature

Gini(Temp)= (4/14) *0.5+(4/14) *0.375+(6/14) *0.445=0.142+0.107+0.190=0.439

c) Now we will calculate Gini Index for Humidity

Humidity	**Yes**	**No**	**No of Instances**
High	3	4	7
Normal	6	1	7

Gini(humidity=high) =1-(3/7)^2-(4/7)^2=0.489

Gini(humidity=normal) =1-(6/7)^2-(1/7)^2=0.244

Now will calculate total Gini Index for humidity feature

Gini(humidity)= (7/14) *0.489+(7/14) *0.244=0.367

d) Now we will calculate Gini Index for Wind

Wind	**Yes**	**No**	**No of Instances**
Weak	6	2	8
Strong	3	3	6

Gini(wind=weak) =1-(6/8)^2-(2/8)^2=0.375

Gini(wind=strong) =1-(3/6)^2-(3/6)^2=0.5

Now will calculate the total Gini Index for wind feature

Gini(wind)= (8/14) *0.375+(6/14) *0.5=0.428

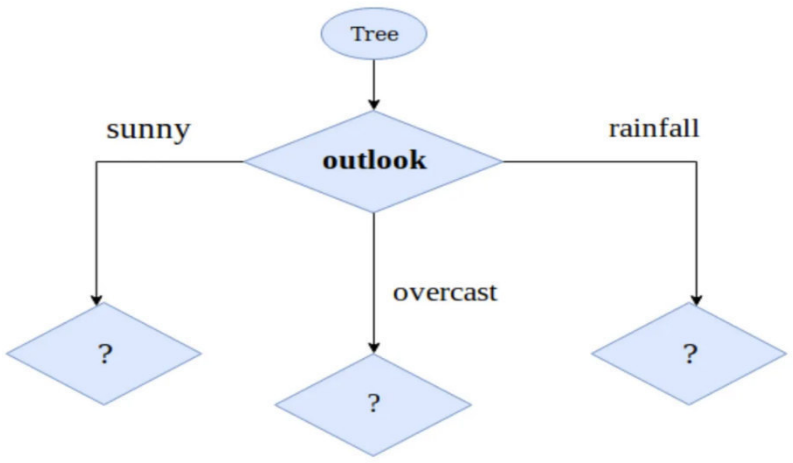
2) Decision for root node

Features	**Gini Index**
Outlook	0.342
Temperature	0.439

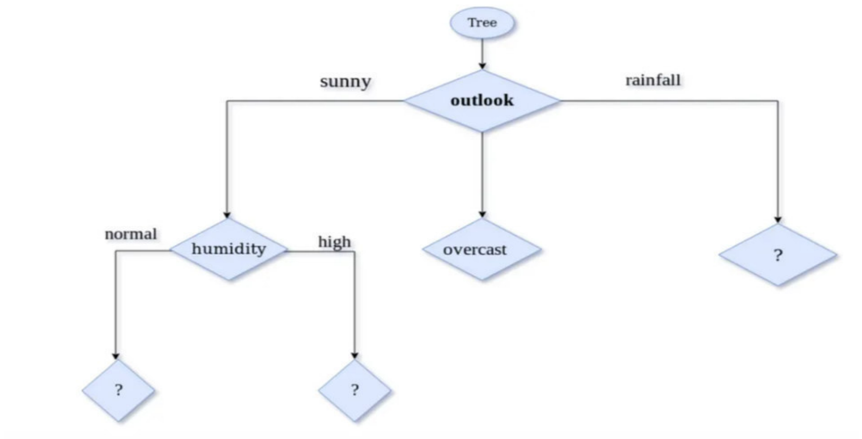
Humidity	0.367
Wind	0.428

From above table, we have seen that Gini index for Outlook is low. Thus outlook will be the root node.

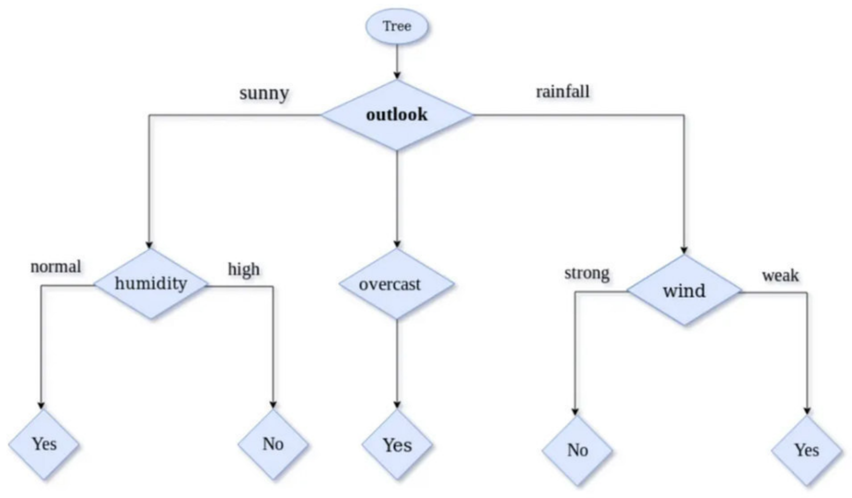
Thus we have



Now the next step is to consider sub data on sunny outlook feature, we need to calculate Gini index for temperature, humidity and wind features respectively. This process will continue and the resulted tree will be like this.



This process will continue and thus the final decision tree will be



STOP TO CONSIDER

It is a greedy algorithm because it chooses the best feature at each stage in the process. It is a variation of the decision tree algorithm. It can handle both classification and regression tasks. In 1984 CART was produced by Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone.

SELF ASKING QUESTION

1. What is the splitting criteria used in cart algorithm?
2. What are the limitations of cart analysis?

CHECK YOUR PROGRESS

6. List down the attribute selection measures used by the ID3 algorithm to construct a Decision Tree.
7. Which should be preferred among Gini impurity and Entropy?

2.6.2 ID3 Algorithm:

Based on the values of the features in the dataset, it splits the dataset into smaller subsets and builds a hierarchical tree in a top-down fashion. It selects the feature that gives the most information about the target variable. The root node represents the entire data set. At each node, the ID3 algorithm selects the attribute that provides the most information gained about the target variable. An attribute that has the highest information gain is selected as a node that separates the data set. Until a halting requirement is satisfied, the procedure continues. Halting conditions like a minimum subset size or a maximum tree depth.

2.6.2.1 Steps of ID3 algorithm:

1. The first step is to select a node(S) as a Root that has the highest Information Gain and low Entropy.

2. It calculates the Entropy and Information gain on each iteration of an algorithm by considering that every node is unused.

3. Select a node based on the Highest Information gain and lowest Entropy.

4. Then Splits set S to produce the subsets of data

5. An algorithm continuously recurs on each subset to make sure that attributes are fresh and Creates the Decision Tree

2.6.2.2 Advantages and disadvantages of ID3

Advantages of ID3

- It is Simple and easy to understand.
- It works well even when training data is less.
- Can work with both discrete and continuous attributes.
- ID3 is less expensive compared to some other complex algorithms.

Disadvantages of ID3

- ID3 tends to create complex trees that may lead to overfitting.
- The presence of a large number of attributes may not be effective.

- Incorrect splits or non-optimal creation of trees may occur due to the presence of Noise or outliers.
- Complex relationships present in the data are difficult to represent using ID3 as it creates a binary tree only.

Problem 2. Consider the following data set. A1 and A2 are two attributes and classification is the target variable. Draw a decision tree using ID3 algorithm.

Instance	**Classification**	**A1**	**A2**
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

Solution:

- In the above data set, we need to find the information gain of A1 and A2. Once we calculate information gain, we can select one of the attributes as the Root node whose information gain is maximum.
- To calculate information gain, we need to calculate entropy. Entropy can be represented as
  S= [3 +,3 -] =1.0 where S is the entropy of the data set since we have 3 +(positive) and 3 –(negative) in the target variable. Value is 1 since an equal number of + and - are in the target variable.
- Now Entropy of each variable A1 and A2 need to calculate.
- First entropy of A1 is
  (a)S_T= [2 +,1 -] meaning we have total three true in A1 and in classification against T of A1 we have two positive (+) and 1 negative (-). Thus
  Entropy(S_T)=-2/3 $\log_2$ 2/3-1/3 $\log_2$ 1/3=0.9183 where 2/3 $\log_2$ 2/3 is the proportion of positive example.
  (b)S_F= [1 +,2 -] meaning we have total three F in A1 and in classification against F we have one positive (+) and two negative (-). Thus
  Entropy(S_F)=-1/3$\log_2$1/3-2/3$\log_2$ 2/3=0.9183

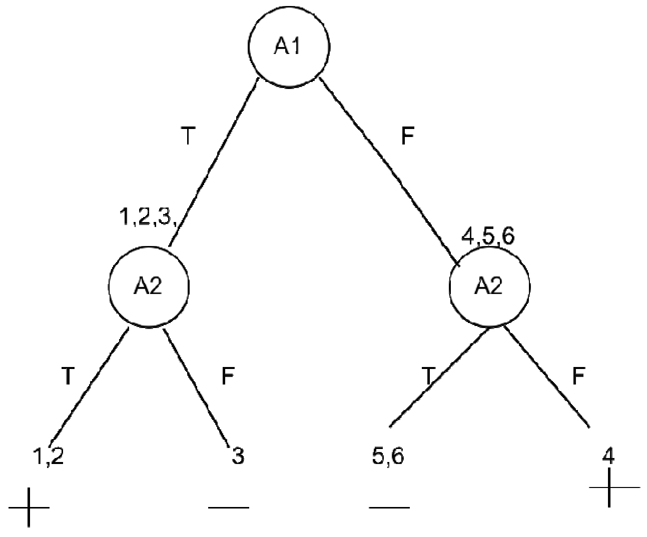
Information gain (S, A1) w.r.t entire dataset is

Gain (S, A1) =Entropy(S)- ∑ (|Sv| / |S|) *Entropy(S)
V€ (T, F)
=Entropy(S)- 3/6*Entropy(S_T) -3/6*Entropy(S_F)
=1.0- 3/6*0.918 -3/6*0.918
=0.0817

- Now we need to calculate the entropy of A2
  (a)Values(A2) =T, F
  S= [3 +,3 -]Entropy(S)=1.0
  S_T= [2 +, 2 -]   Entropy(S_T)=1.0
  S_F= [1 +, 1 -]   Entropy(S_F)=1.0

  Gain (S, A2) = Entropy(S) - ∑ (|Vs.| / |S|) *Entropy(S)
  V€ (T, F)
  = Entropy(S)- 4/6*Entropy(S_T) -2/6*Entropy(S_F)
  = 1.0 – 4/6*1.0- 2/6 * 1.0= 0.0
  Gain (S, A1) =.817
  Gain (S, A2) =0.0
  Since the Information gain of A1 is maximum, so A1 will be the root node.

  The decision tree using ID3 will be like this:



STOP TO CONSIDER

ID3 stands for Iterative Dichotomiser 3 and it is so called such because at each step the algorithm repeatedly divides features into two or more groups.

SELF ASKING QUESTIONS

1. What is the splitting criteria of ID3 algorithm?
2. What are the characteristics of ID3 algorithm?

CHECK YOUR PROGRESS

8. Draw a decision tree using ID3

Age	Income	Student	Credit_rating	Buys_Computer
<=30	High	No	Fair	no
<=30	High	No	Excellent	no
31..40	High	No	Fair	yes
>40	Medium	No	Fair	yes
>40	Low	Yes	Fair	yes
>40	Low	Yes	Excellent	no
31…40	Low	Yes	Excellent	Yes
<=30	Medium	No	Fair	No
<=30	Low	Yes	Fair	Yes
>40	Medium	Yes	Fair	Yes
<=30	Medium	Yes	Excellent	Yes
31…40	Medium	No	Excellent	Yes
31…40	High	Yes	Fair	Yes
>40	Medium	No	Excellent	yes

2.6.3 C4.5 Algorithm

C4.5algorithms is an extension of ID3.It is developed by Ross Quinlan.It is mainly used for classification. It creates a decision tree for both discrete and continuous attributes that's why it is considered to be an effective technique. By utilizing of gain ratio and decreased error pruning, the model's accuracy is increased and overfitting is prevented.Whenthere arealot of features in the dataset, it might not function effectively.

C4.5 sorts the attribute's values first, when dealing with continuous attributes,and then chooses the midpoint between each pair of adjacent values as a potential split point. Next, it calculates the information gain or gain ratio and the attributes that have the highest information gain will be the split point.

2.6.3.1 Steps of C4.5 Algorithm

1. First we need to compute Entropy_info for the whole dataset based on the target variable.

   $$\text{Entropy\_info}(T) = -\sum_{i=1}^{m} P_i \log P_i$$

2. Next for each of the attribute in the training dataset,we need to compute Entropy_info,

   $$\text{Entropy\_info}(T,A) = \sum_{i=1}^{v} |A_i| / |T| * \text{Entropy\_info}(A_i)$$

   Information_Gain(A)=Entropy_Info(T) – Entropy_Info (T, A)

   $$\text{Split\_Info}(T,A) = -\sum_{i=1}^{v} |A_i|/|T| * \log_2 |Ai|/|T|$$

   Gain_ratio(A)= Information_gain(A)/Split_infor(T,A)

3. Choose the attribute with the highest Information_gain as best split attribute.
4. The best-split attribute will be considered as root node.
5. The root node branced into subtrees with each subtree considered to be outcome of the test of the root node attribute.
6. Recursively need to repeat the above step until no more training instances are available or a leaf node is derived.

2.6.3.2 Advantages and Disadvantages of C4.5

Advantages of C4.5 Algorithm:

1. It handles both discrete and continuous values.
2. It handles training data with missing attribute values by marking it as '?' but these missing values are not considered in the calculations.
3. Pruning trees after creation.
4. It improves computational efficiency.

Disadvantages of C4.5 Algorithm:

1. When the training data set is small, it does not work well.
2. Decision tree changes when there is a small change in data takes place (especially when the variables are close to each other).

SELF ASKING QUESTIONS

1. Is C4.5 successors of ID3?
2. What is the advantage of C4.5 over ID3 algorithm?
3. What is the difference between CART and C4.5 algorithms?

CHECK YOUR PROGRESS

9. What is C4.5 used to build?

10. What is the new features of C4.5?

.

2.7 Summing Up

This unit tells us about the decision tree. How a data set can be represented in the form of a decision tree and the various types of decision trees. It tells about CART, ID3, and C4.5 algorithms. How the root node is selected in each case is explained? Also explained about the concept of entropy, information gain, and information ratio that is required to split the dataset.

2.8 Answer to Check Your Progress

ANSWER 1. The reduction in entropy is nothing but information gain which isdue to the selection of a particular attribute. Information gain biases the Decision Tree against considering attributes with a large number of distinct values, which might lead to overfitting.

The **information Gain Ratio** is used to solve this problem.

ANSWER 2. Decision Trees are suitable for the following cases:

1. When the data are in the form of tabular, decision trees are most suitable in that case.
2. When the outputs are discrete.
3. Explanations for Decisions are required.
4. When the training data may contain errors and noisy data(outliers).
5. The training data may contain missing feature values.

ANSWER 3. The following are popular algorithms for constructing decision trees:

1. ID3 (Iterative Dichotomiser): Uses Information Gain as an attribute selection measure.
2. C4.5 (Successor of ID3): Uses Gain Ratio as an attribute selection measure.
3. CART (Classification algorithm and Regression Trees) – Uses Gini Index as an attribute selection measure.

ANSWER 4. Root node, Decision node, and Leaf node.

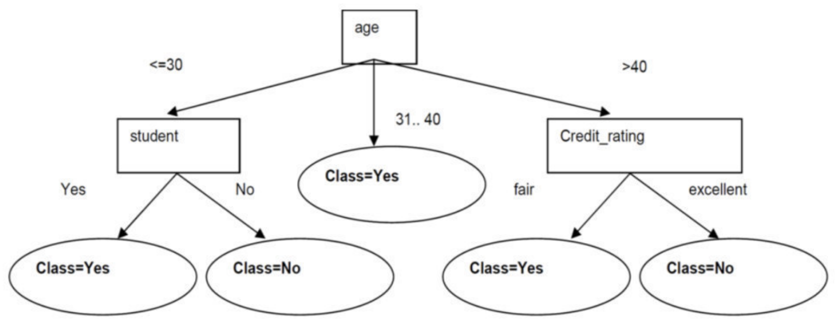
ANSWER 5. The difference between the entropy of a data segment before and after the split is Information gain, i.e., reduction in impurity due to the selection of an attribute.

The higher the difference, the higher the information gain, and it implies lower entropy and thus better the feature used for the split.

ANSWER 6. Entropy, Information Gain

ANSWER 7. Gini impurity tends to isolate the most frequent class in its own branch of the Tree, while entropy tends to produce slightly more balanced Trees.

ANSWER 8.



Decision Tree for Buys Computer

ANSWER 9. C4.5 is used to build a decision tree from the training data set using the concept of information entropy.

ANSWER 10. C4. 5 algorithm has many additional features, including handling missing values, decision tree pruning, continuous attribute value ranges, derivation of rules etc.

2.9 POSSIBLE QUESTIONS

1. What is the use of CART algorithms?
2. What are ID3 algorithms in decision trees?
3. What is entropy in decision trees?

4. What is Information gained in decision trees?
5. What is gini entropy in decision trees?
6. What are root and leaf nodes in decision trees?
7. What kind of problem decision trees are most suitable?
8. How does a decision tree handle missing attribute values?
9. Why are decision trees unstable?
10. Mention some algorithms for deriving decision trees.

2.10 References and Suggested Reading

1.https://ebooks.inflibnet.ac.in/csp15/chapter/decision-tree-algorithm-id3/

2. Data Mining with Decision Trees (Theory and Applications) By Lior Rokach and Oded Maimon

3.https://towardsdatascience.com/decision-tree-questions-to-ace-your-next-data-science-interview-692ee246b6ae

---×---

UNIT: 3
GENETIC ALGORITHMS

Unit Structure

3.1 Introduction

In 1975, John Holland had proposed Genetic algorithms at the University of Michigan. The concept of Genetic algorithms is based on the process of biological evolution. Genetic algorithms are used to solve computational problems by developing a simulation of natural evolution on computer systems. In general, Genetic algorithms are used to solve complex optimization problems. In this unit, the concept of Genetic algorithms will be discussed. The use of Genetic algorithm in data mining will also be presented in this unit.

3.2 Objectives

After going through this chapter, we will be able to learn:

- About the concept of Genetic algorithms.
- About different steps and key terms in Genetic algorithms.
- About the genetic operators.
- About the use of Genetic algorithms in Data Mining.
- About advantages and disadvantages of Genetic algorithms.

3.3 Introduction to Genetic Algorithm

In general, Genetic algorithm is a searching technique to provide efficient solution to complex optimization problems. It is based on the Darwin's principle, 'survival of the fittest'. Genetic algorithm is the way to simulate the natural evolution on the computer system to solve complex problems. Different important key terms associated with Genetic algorithm are defined in the following points.

- **Gene:** A gene represents a particular variable or a property. In case of Genetic algorithms, genes can be defined as elementary unit of information or decision variables that participate in the solution of a problem. The value of a gene is termed as an allele. For example: In case of binary encoding, a gene is either 0 or 1.
- **Chromosome:** Achromo some is a group of genes. For example: In case of binary encoding, a string of 0's and 1's like '1001110' is a chromosome.
- **Population:** A population is a group of chromosomes which are the possible solutions of a problem.
- **Fitness function:** A fitness function is a function that used to estimate a fitness score for each of the possible solutions available in a population. Each solution available in a population is allocated with a fitness score estimated by a fitness function so that its appropriateness and capability to solve the corresponding problem can be identified. So a fitness function provides the foundation for the selection of an optimal solution to the corresponding problem. It also helps to select the suitable solutions available in a population which are going to be used for the evolution of that

population. In this process of evolution, quality of the possible solutions becomes better.

In general, an efficient fitness function holds the following characteristics.

- An efficient fitness function is fast enough to estimate the fitness scores of any number of possible solutions.
- An efficient fitness function can handle multiple qualities of possible solutions to estimate fitness scores.
- An efficient fitness function can estimate effective fitness scores that correctly fulfil the objectives of the corresponding problem and accurately identify the effectiveness of the possible solutions without ambiguity.
- An efficient fitness function also capable of excluding the solutions that is not useful as a feasible solution to the corresponding problem.

- **Selection:** Selection is the procedure in Genetic algorithms to select the solutions from the current population which will be used as parents to develop the next generation solutions. In this process, solutions that hold higher fitness scores have the higher probability to be selected as parents for the next generation.
- **Genetic operator:** Genetic operators are the mechanisms that used to develop new solutions from the existing solutions available in a population. So, in Genetic algorithms, genetic operators are used in the process of population evolutions so that optimal solution can be achieved for the corresponding problems.

3.4 Steps in Genetic Algorithm

Different steps in Genetic algorithm are discussed in the following points.

- **Step 1:** An initial population of possible solutions is produced for the corresponding problem in the first step of

Genetic algorithm. In general the size of the initial population is predefined. The possible solutions for the corresponding problem to construct the population are generated randomly.

- **Step 2:** An effective fitness function is used to estimate the fitness scores of each possible solution available in the current population. The fitness score of a solution indicates the effectiveness or quality of the solution for the corresponding problem.
- **Step 3:** Selection procedure is performed to select effective possible solutions depending upon their fitness scores from the population so that these solutions can be used to develop new more efficient solutions. In general, it is a probabilistic procedure. The probability for selecting a solution will be high if its fitness score is high.
- **Step 4:** Genetic operations are performed to develop a new more efficient population for the corresponding problem by using the selected possible solutions achieved in step 3.
- **Step 5:** From step 2 to step 4 are repeated to develop more efficient new generation population than the earlier one until the near optimal solution for the corresponding problem is achieved. This population evolution process can be stopped depending upon the termination condition applied in the algorithm to solve the corresponding problem. Some of the general termination conditions are presented in the following points.
    - ➢ Population evolution process can be stopped after a predefined number of population evolutions or a predefined amount of time.
    - ➢ A threshold value can be defined to stop the population evolution process. When the fitness score of any solution become greater than the threshold value then the population evolution process can be stopped.
    - ➢ If it is realized that there will be no better possible solutions can be evolved then the population evolution process can be stopped.

In the population evolution process, older solutions are replaced by the newly developed solutions in the population. In general, two solution replacement approaches may be implemented depending upon the corresponding problem to generate the next generation population. All older solutions may be replaced by the newly developed solutions in a population. In the second approach, only less effective solutions may be replaced by the newly developed effective solutions.

3.5 Types of Genetic Operators

We have already learnt about the definition of Genetic operators in the section 16.3. In this section we are going to discuss the different types of genetic operators. In general, two types of Genetic operators are available in Genetic algorithm. These two Genetic operators are Crossover and Mutation.

3.5.1 Crossover

Crossover is a genetic operator in Genetic algorithm. In this approach, one or more new chromosomes are created by combining genetic information of two existing chromosomes. In this process, a crossover point has been randomly selected on the two chromosomes that are selected as parent chromosomes. Then new child chromosome is produced by swapping the parts beyond the crossover points between the parent chromosomes. For example, let us consider two parent chromosomes that are '1100110' and '1001110'. The crossover point for the first parent chromosome, '1100110' is selected after the segment, '1100' and the crossover point for the second parent chromosome, '1001110' is selected after the segment, '1001'. Then after the crossover procedure, the new child chromosome will be '1100110'.In this process, '1100' from the first parent chromosome is combined with '110' from the second parent chromosome.

3.5.2 Mutation

Mutation is another genetic operator in Genetic algorithm. In this approach, new chromosomes are constructed by altering small randomly selected portions of the existing chromosomes. So, the structure of the newly developed chromosomes may not be related to the existing chromosomes in the corresponding population. As a result, mutation can provide a next generation population with newly developed solutions from the other parts of the search space. For example, let us consider an existing chromosome, '11011001'. Now a new chromosome can be developed by altering the last '1' in the mentioned chromosome to '0'. So the new chromosome will be '11011000'.A new chromosome can also be developed by swapping two bits in the mentioned chromosome.

3.6 Genetic Algorithms for Data Mining

Genetic algorithm is also a useful tool for data mining. Some of the general applications of genetic algorithm in the field of data mining are presented in the following points.

- In data mining, Genetic algorithm may be used for the hypothesis evaluation and its improvement. In this process different fitness functions can be used to evaluate the fitness scores of the possible candidates for the hypothesis improvement.
- The performance of an existing data mining technique can be improved by applying Genetic algorithm with it. In this process, a hybrid technique is developed by combining Genetic algorithm with an existing data mining technique. For example: Genetic algorithm can be used with the neural networks. Genetic algorithm can also be used with the existing greedy search algorithm to develop optimal decision tree. In the process of decision tree induction, Genetic algorithm can be used for population evolution so that an optimal tree can be built.
- Genetic algorithm can be used in Rule mining to search appropriate reliable association rules that can be utilized in the solution of optimization problems. In this process, each chromosome in the corresponding population is an association rule. Genetic algorithm can also be utilized to optimize the association rules that are created by the Apriori algorithm.
- Genetic algorithm may be used to choose the appropriate features from a large group of features related to the corresponding problem so that the performance of the system to solve the problem can be improved. In this process, a chromosome is a subset of features. The fitness function estimates the fitness score of each chromosome

based on the system performance when the chromosome will be used to solve the corresponding problem.

- Genetic algorithm can be used in classification by identifying most relevant rules and features to categorize data into different significant groups. In this process, the fitness function may estimate the fitness scores based on the correctness, simplicity or coverage of the classification. For example, Genetic algorithm may be used in Text classification, Image classification etc.
- Genetic algorithm can be utilized in clustering techniques to find out suitable cluster centres in the dataset consisting of features so that optimized clusters can be generated. As a result the performance of the clustering technique may be improved.

3.7 Advantages and Disadvantages of Genetic Algorithms

We have already learnt that Genetic algorithm is a useful tool to solve complex optimization and searching problems. Different general advantages of Genetic algorithm are presented in the following points.

- Genetic algorithms always continue to search for near optimal solution for the corresponding problem.
- Genetic algorithms can provide near optimal solutions to large and complex problems. In case of complex optimization problems, Genetic algorithm may provide better global optimal solution than the traditional techniques.
- Genetic algorithms can be easily combined with the traditional data mining techniques to provide better performance in case of different data mining problems like clustering, classification, optimization of association rules etc.
- Genetic algorithms can be applied in diverse areas like pattern recognition, robotics, economics, biotechnology etc.
- In case of a new problem, Genetic algorithm may provide correct solutions. More accurate results for the new area can

be produced by other specialized techniques after acquiring a good knowledge about it.

- Genetic algorithms are adaptive enough such that they may provide better results with dirty data and missing values than the other techniques.
- Genetic algorithms can provide better scalability in case of parallel processing environments.

Different disadvantages or limitations are also associated with Genetic algorithms. The general disadvantages or limitations are presented in the following points.

- Genetic algorithms are computationally costly. It is hard to understand the concept of Genetic algorithms.
- Genetic algorithms can be more time consuming than the traditional techniques.
- It is a difficult job to find out a perfect fitness function for a Genetic algorithm to solve the corresponding problem.
- Development of a most relevant initial population to solve the corresponding problem is a difficult task.
- We already know that the performance of a Genetic algorithm is dependent on the genetic operators which are crossover and mutation. Now, it is a difficult job to decide how efficient crossover and mutation will be implemented for the corresponding problem.
- Different parameters like population size, crossover rate, mutation rate etc. must be efficiently determined so that Genetic algorithms can provide near optimal solutions.
- Fitness function of Genetic algorithms may require a good amount of raw data so that it can estimate appropriate fitness score for each of the chromosomes.

STOP TO CONSIDER

The data that achieved directly from the source is referred as raw data. Raw data is also termed as unprocessed data as it is not cleaned or processed after collection from the source.

Check Your Progress

1. Fill in the blanks

(a) _______is a function that used to estimate a fitness score for each of the chromosomes available in a population.

(b) _____is a group of chromosomes which are the possible solutions of a problem.

(c) _____are the mechanisms that used to develop new solutions from the existing solutions available in a population.

(d) ______is a technique to simulate the natural evolution on the computer system to solve complex problems.

(e) In data mining, Genetic algorithm can be used to solve_____ problems.

2. Choose the correct option

(a) Which of the following is not a genetic operator?
(i) Crossover
(ii) Chromosome
(iii) Mutation
(iv) None of the above

(b) Which of the following is performed at the first step in Genetic algorithm?
(i) Crossover
(ii) Development of an initial population.
(iii) Mutation
(iv) Both (i) and (iii)

(c) Which of the following is an advantage of Genetic algorithm?
(i) Genetic algorithms always continue to search for near optimal solution for the corresponding problem.
(ii) Genetic algorithms are always computationally less costly.
(iii) Genetic algorithms are always faster than the traditional techniques.
(iv) All of the above.

(d) Which of the following is a disadvantage of Genetic algorithms?

(i) Genetic algorithms are difficult to understand.
(ii) Genetic algorithms may be slower than the traditional techniques.
(iii) Genetic algorithms are computationally costly.
(iv) All of the above.

(e) Which of the following is true in case of mutation?
(i) It is a fitness function in Genetic algorithm.
(ii)New chromosomes are constructed by altering small randomly selected portions of the existing chromosomes.
(iii) One or more new chromosomes are created by combining genetic information of two existing chromosomes.
(iv) None of the above.

3.8 Summing Up

- The concept of Genetic algorithm is based on the Darwin's principle, 'survival of the fittest'. Genetic algorithm is a technique to simulate the natural evolution on the computer system to solve complex problems.
- A gene represents a particular variable or a property. A chromosome is a group of genes and a population is a group of chromosomes that are the possible solutions of a problem.
- Genetic operators are the mechanisms that used to create new chromosomes from the existing chromosomes available in a population.
- A fitness function is a function that used to estimate a fitness score for each of chromosomes available in a population.
- Selection is the procedure in Genetic algorithms to select the solutions from the current population which will be used as parents to develop the next generation solutions.
- Steps in Genetic algorithm are:
  1. Development of an initial population.
  2. Application of an effective fitness function to estimate the fitness scores of each possible solution.
  3. Perform selection procedure to select effective possible solutions that can be used to develop new more efficient solutions.

4. Genetic operations are performed to develop a new more efficient population by using the selected solutions.
5. From step 2 to step 4 are repeated to develop more efficient new generation population than the earlier one until the near optimal solution for the corresponding problem is achieved.

- Two Genetic operators in Genetic algorithm are Crossover and Mutation. In crossover, one or more new chromosomes are created by combining genetic information of two existing chromosomes. In mutation, new chromosomes are constructed by altering small randomly selected portions of the existing chromosomes.
- Genetic algorithms can be easily combined with the traditional data mining techniques to provide better performance in case of different data mining problems like clustering, classification, optimization of association rules etc.
- Genetic algorithms are computationally costly and it is hard to understand its concept.
- Genetic algorithms can be more time consuming than the traditional techniques.
- It is a difficult job to find out a perfect fitness function for a Genetic algorithm.
- Development of a most relevant initial population for a Genetic algorithm is a difficult task.
- Different parameters like population size, crossover rate, mutation rate etc. must be efficiently determined so that Genetic algorithms can provide near optimal solutions.
- Fitness function of Genetic algorithms may require a good amount of raw data so that it can estimate appropriate fitness score for each of the chromosomes.

3.9 Answers to Check Your Progress

1.

(a) A fitness function.
(b) A population.
(c) Genetic operators.
(d) Genetic algorithm
(e) Clustering

2.

(a)(ii) Chromosome

(b)(ii) Development of an initial population.

(c)(i) Genetic algorithms always continue to search for near optimal solution for the corresponding problem.

(d)(iv) All of the above.

(e)(ii)New chromosomes are constructed by altering small randomly selected portions of the existing chromosomes.

3.10 Possible Questions

1. What is Genetic algorithm?
2. What is fitness function?
3. Write down the different steps in Genetic algorithms.
4. Explain crossover and mutation.
5. Write down the applications of Genetic algorithm in Data mining.
6. Write down the advantages and disadvantages of Genetic algorithms.

3.11 References and Suggested Readings

1. Berson, A., & Smith, S. J. (1997). Data warehousing, data mining, and OLAP. McGraw-Hill, Inc..
2. Margaret, H., Sridhar. S (2006). Data Mining-"Introductory and Advanced Concepts". Pearson Education India.
3. Inmon, W. H. (2005). *Building the data warehouse*. John wiley & sons.
4. Ponniah, P. (2004). *Data warehousing fundamentals: a comprehensive guide for IT professionals*. John Wiley & Sons.
5. Pujari, A. K. (2001). *Data mining techniques*. Universities press.
6. Adriaans, P., Zantinge. D. (1996). *Data mining*. Pearson Education India.

---×---

UNIT: 4
ARTIFICIAL NEURAL NETWORK

Unit Structure:

4.1 Introduction

In this chapter, we delve into the fascinating world of neural networks, focusing on their applications in feature extraction and clustering. We begin by exploring how neural networks can automatically discover and extract meaningful features from raw data, transforming it into a more useful format for various tasks. Next, we examine neural networks designed specifically for clustering, highlighting their ability to group similar data points without predefined labels. Additionally, we discuss different neural network architectures, including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), and their roles in these processes. Through practical examples and detailed explanations, we aim to provide a comprehensive understanding of how these advanced techniques work and their significant advantages and disadvantages. By the end of this chapter, readers will have a solid grasp of the concepts and applications of neural networks in feature extraction and clustering, ready to apply these powerful tools to their own data-driven challenges.

4.2 Unit Objective

By the end of this unit, learners will be able to:

- ✓ Understand the role and mechanisms of neural networks in feature extraction and clustering.
- ✓ Identify and describe various neural network architectures used for these tasks.
- ✓ Apply neural networks to automatically extract features from raw data.
- ✓ Utilize neural networks for clustering data without predefined labels.
- ✓ Evaluate the advantages and disadvantages of using neural networks in these contexts.

4.3 Neural Network: Importance and Applications in Modern Machine Learning

Neural networks are key to artificial intelligence (AI) and machine learning (ML), inspired by the human brain. These models recognize patterns, make decisions, and learn from data. They have advanced significantly, now used in image and speech recognition, autonomous vehicles, and natural language processing.

Neural networks consist of layers of nodes (neurons) performing simple computations. They include an input layer, hidden layers, and an output layer. Connections between nodes have weights that adjust during training to improve accuracy.

The concept dates back to the 1940s, but major progress occurred in the 1980s and 1990s with backpropagation, which trains networks by adjusting weights to minimize errors. This process is crucial for deep learning.

Neural networks impact healthcare, finance, entertainment, and transportation, aiding in disease diagnosis, algorithmic trading, content recommendation, and autonomous driving. As research advances, neural networks continue to revolutionize technology and science.

What is Neural Network?

A neural network is a method in artificial intelligence that teaches computers to process data in a way that is inspired by the human brain. A neural network is an artificial system made of interconnected nodes (neurons) that process information, modeled after the structure of the human brain. It is employed in machine learning jobs where patterns are extracted from data. Therefore neural network is a type of machine learning process, called deep learning that uses interconnected nodes or neurons in a layered structure that resembles the human brain. It creates an adaptive system that computers use to learn from their mistakes and improve continuously. Thus, artificial neural networks attempt to solve complicated problems, like summarizing documents or recognizing faces, with greater accuracy.

Importance of Neural Network

Neural networks are crucial for recognizing patterns, solving complex problems, and adapting to dynamic environments. Their ability to learn from data has profound implications, transforming technologies such as natural language processing and autonomous vehicles, while also automating decision-making processes and boosting efficiency across various sectors. The advancement of artificial intelligence relies heavily on neural networks, which propel innovation and shape the future of technology.

Fundamental difference between traditional computers and neural networks is the way in which they function.

Traditional Computing	Neural Networks
• Deductive Reasoning: We can apply known rules to input data to produce output	• Inductive Reasoning: Given input and output data (training examples), we construct the rules
• Computation is centralized	• Computation is collective
• Memory is packetted, literally stored and location	• Memory is distributed, internalized and content

addressable	addressable
• Not fault tolerant	• Fault tolerant
• Exact	• Inexact
• Static Connectivity	• Dynamic connectivity
• Applicable if well defined rules with precise input data	• Applicable if rules are unknown or complicated, or if data is noisy or partial

Table 1: Traditional Computing Vs Neural Network

<u>How Neural Network works?</u>

Let's understand how a neural network functions with an example:

Imagine a neural network designed for classifying emails. The input layer receives features such as the email content, sender information, and subject line. These inputs are processed by being multiplied by adjustable weights and passed through hidden layers. During training, the network learns to identify patterns that suggest whether an email is spam. The output layer, using a binary activation function, predicts whether the email is spam (1) or not (0). Through iterative weight adjustments via backpropagation, the network improves its ability to differentiate between spam and legitimate emails, demonstrating the practical application of neural networks in tasks like email filtering.

Neural networks are like complex brain-like systems. They have an input layer, one or more hidden layers, and an output layer, all made up of connected artificial neurons. The basic process includes two main steps: forward propagation and backpropagation.

Training a neural network involves both forward and backward propagation. In machine learning and data mining, backpropagation is used to enhance prediction accuracy by calculating derivatives through backward propagation. This process involves moving from the output layer (right) to the input layer (left). Conversely, forward propagation moves data from the input layer (left) to the output layer (right).

A neural network can be viewed as a network of connected input/output nodes. The accuracy of each node is represented by a loss function or error rate. Backpropagation determines the slope of the loss function concerning the other weights in the neural network.

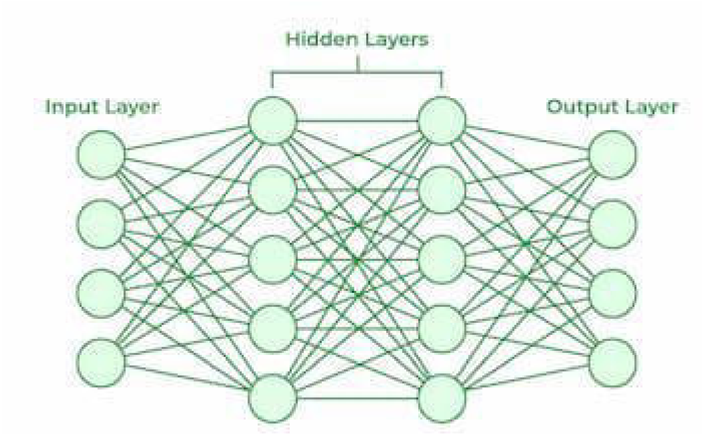


Fig (a): Working Principle of Neural Network

<u>Applications of Neural Network</u>

Neural networks have a wide range of applications across various industries, including:

- Medical diagnosis through medical image classification
- Targeted marketing via social network filtering and behavioural data analysis
- Financial predictions by analyzing historical financial data
- Electrical load and energy demand forecasting
- Process and quality control
- Chemical compound identification

Here are four key applications of neural networks:

Computer Vision: Computer vision enables computers to interpret and analyze visual data from images and videos. Neural networks allow computers to recognize and distinguish images much like humans do. Some applications include:

- Visual recognition in self-driving cars to identify road signs and other road users
- Content moderation to automatically remove unsafe or inappropriate content from visual media
- Facial recognition to identify and assess facial attributes such as open eyes, glasses, and facial hair
- Image labelling to detect brand logos, clothing, safety gear, and other details

Speech Recognition: Neural networks can process human speech despite differences in patterns, pitch, tone, language, and accent. Speech recognition is used in:

- Virtual assistants like Amazon Alexa
- Automatic transcription software for tasks such as:
- Assisting call centre agents and classifying calls
- Converting clinical conversations into real-time documentation
- Accurately subtitling videos and meeting recordings

Natural Language Processing (NLP):NLP allows computers to understand and interpret human-created text. Neural networks enhance NLP capabilities, leading to applications such as:

- Automated virtual agents and chatbots
- Automatic organization and classification of written data
- Business intelligence analysis of documents like emails and forms
- Sentiment analysis by indexing key phrases in social media comments
- Document summarization and article generation

Recommendation Engines:Neural networks can analyze user activity to provide personalized recommendations. They can also discover new products or services that may interest a user. For example, Curalate, a startup in Philadelphia, helps brands turn social media posts into sales. Using neural networks, their intelligent product tagging (IPT) service automates the collection and curation of user-generated social content. IPT identifies and recommends products based on the user's social media activity, making it easier for consumers to find and purchase products without searching through online catalogs.

Advantages of Neural Networks:

Non-linearity: Neural networks can model complex relationships between inputs and outputs, allowing them to learn intricate patterns in data that linear models may miss.

Adaptability: They are highly adaptable to different types of data and tasks, including classification, regression, clustering, and pattern recognition. With proper architecture and training, neural networks can be tailored to various applications.

Feature Learning: Neural networks can automatically learn relevant features from raw data, eliminating the need for manual feature engineering in many cases. This ability is particularly useful when dealing with high-dimensional and unstructured data such as images, text, and audio.

Parallel Processing: Modern neural network libraries and frameworks leverage parallel processing on GPUs and specialized hardware, enabling efficient training and inference on large datasets.

Robustness to Noise: Neural networks can handle noisy data and generalization to unseen examples by learning from a diverse set of examples during training.

Scalability: They can scale to large datasets and complex problems by adding more layers, neurons, or using advanced architectures like convolutional neural networks (CNNs) and recurrent neural networks (RNNs).

Disadvantages of Neural Networks:

Complexity: Designing, training, and optimizing neural networks can be complex and time-consuming, requiring expertise in architecture selection, hyperparameter tuning, and optimization techniques.

Overfitting: Neural networks are prone to overfitting, where the model learns to memorize the training data rather than generalize to unseen data. Techniques such as dropout, regularization, and cross-validation are used to mitigate overfitting.

Black Box Nature: The internal workings of neural networks can be challenging to interpret, making it difficult to understand how they arrive at their predictions or decisions. This lack of interpretability may be problematic in applications where transparency is crucial, such as healthcare and finance.

Data Requirements: Neural networks often require large amounts of labelled data for training, which may be costly or time-consuming to obtain. Additionally, they may not perform well on tasks with limited training data or in domains where labelled data is scarce.

Computationally Intensive: Training complex neural networks with many layers and parameters can be computationally intensive, requiring powerful hardware such as GPUs or specialized accelerators. This computational cost may limit their practicality for certain applications, especially in resource-constrained environments.

Hyperparameter Sensitivity: Neural networks have numerous hyperparameters, such as learning rate, batch size, and network architecture, which need to be carefully tuned to achieve optimal performance. Finding the right combination of hyperparameters can be challenging and often requires extensive experimentation.

4.4 Types of Neural Network

Different types of Neural Networks are:

- Feedforward Neural Network (FNN)
- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Long Short-Term Memory (LSTM) Network
- Gated Recurrent Unit (GRU) Network
- Autoencoder
- Generative Adversarial Network (GAN)
- Radial Basis Function (RBF) Network
- Modular Neural Network (MNN)
- Spiking Neural Network (SNN)

4.4.1 Feedforward Neural Network

A Feedforward Neural Network (FNN) is the simplest type of artificial neural network architecture where the data flows in one direction—from the input layer, through one or more hidden layers, to the output layer. Unlike other types of neural networks, such as recurrent neural networks, FNNs do not have loops or cycles, meaning there is no backtracking of information.

Structure of Feedforward Neural Networks:

Input Layer: This is where the model receives its initial data. Each node in this layer represents an input feature.

Hidden Layers: These are the intermediary layers where computations are performed. Each node in a hidden layer takes inputs from the previous layer, applies a weighted sum, and passes the result through an activation function.

Output Layer: This layer produces the final output of the network, representing the predictions or classifications based on the processed input data.

Working of Feedforward Neural Networks

Initialization: The network begins with initial random weights assigned to the connections between nodes.

Forward Propagation: Data is passed from the input layer, through the hidden layers, and finally to the output layer. At each node, the input data is multiplied by the weights, summed, and passed through an activation function to introduce non-linearity.

Prediction: The output layer provides the final output based on the computations done in the hidden layers.

Training: The network is trained using a labelled dataset. During training, the weights are adjusted to minimize the error in predictions, usually measured by a loss function. This process often uses back propagation, although FNNs themselves do not inherently include the back propagation process.

Example of Feedforward Neural Network

Consider a simple FNN designed to classify images of handwritten digits (0-9) from the MNIST dataset, which consists of 28x28 pixel greyscale images.

Advantages of Feedforward Neural Networks (FNN)

- **Simplicity and Ease of Implementation:** FNNs have a straightforward architecture with a clear flow of data from input to output, making them easier to understand and implement compared to more complex neural network types.
- **Effective for a Wide Range of Problems:** They can handle various tasks, including regression, classification, and pattern recognition, making them versatile in different domains such as image and speech recognition.
- **Deterministic Output:** Given a specific input, FNNs always produce the same output, which is beneficial for tasks requiring consistent and predictable results.

- **No Issues with Gradient Vanishing or Exploding:** Unlike recurrent neural networks (RNNs), FNNs do not suffer from gradient vanishing or exploding problems because the data flows in one direction without loops.
- **Easier to Train:** Training FNNs is often simpler and faster, particularly for shallow networks, as there are no temporal dependencies to consider.
- **Parallelization:** The feed forward nature of these networks allows for easy parallelization of computations, making them efficient to run on modern hardware, such as GPUs.

Disadvantages of Feedforward Neural Networks (FNN)

- **Limited to Static Input Data:** FNNs are not well-suited for handling sequential or time-series data because they do not have a mechanism to retain information from previous inputs. Tasks like language modeling or time-series prediction often require recurrent or other specialized neural networks.
- **Requires Large Amounts of Data:** Like most neural networks, FNNs require a significant amount of labeled training data to achieve good performance, which can be a limitation in data-scarce domains.
- **Lack of Temporal Dynamics:** FNNs cannot capture temporal dependencies or dynamics, making them unsuitable for tasks where the order of inputs matters, such as video analysis or speech recognition.
- **Over fitting Risk:** Without proper regularization techniques (like dropout, L2 regularization, etc.), FNNs can easily overfit to the training data, especially when the network is deep or the dataset is small.
- **Computationally Intensive for Large Networks:** For deep networks with many hidden layers and nodes, the computational cost of training can be high, requiring significant processing power and memory.
- **Parameter Sensitivity:** The performance of FNNs is sensitive to the choice of hyperparameters, such as learning rate, number of hidden layers, and number of neurons per layer. Finding the optimal set of hyperparameters often requires extensive experimentation and cross-validation.

- **Black Box Nature:** Despite their simplicity, FNNs still operate as black boxes to some extent, making it challenging to interpret and understand the internal workings and decisions of the network.

4.4.2 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a specialized type of artificial neural network designed primarily for processing structured grid data, such as images. They have proven exceptionally effective in tasks like image and video recognition, medical image analysis, and even in natural language processing.

Structure of Convolutional Neural Networks

Input Layer: The input layer receives raw pixel data from an image, typically structured as a 2D grid (for grayscale images) or a 3D grid (for color images with RGB channels).

Convolutional Layers: These layers apply convolutional filters (or kernels) to the input. Each filter slides over the input data, performing element-wise multiplications and summations, producing a feature map. Convolutional layers help in detecting local patterns such as edges, textures, and simple shapes.

Activation Function: After each convolution, an activation function like ReLU (Rectified Linear Unit) is applied to introduce non-linearity into the model, allowing it to learn more complex patterns.

Pooling Layers: Pooling layers reduce the spatial dimensions of the feature maps. The most common pooling operation is max pooling, which takes the maximum value from a small region of the feature map. Pooling helps in making the model invariant to small translations and reduces the computational load.

Fully Connected Layers: After several convolutional and pooling layers, the high-level reasoning is done via fully connected layers. These layers take the flattened feature maps as input and output a final classification score or regression value.

Output Layer: The final layer, often a softmax layer for classification tasks, provides the probability distribution over the possible classes.

Example of Convolutional Neural Network

Let's consider a simple example of a Convolutional Neural Network (CNN) designed to recognize handwritten digits from the MNIST dataset. The MNIST dataset consists of 28x28 pixel grayscale images of handwritten digits (0-9).

Advantages of Convolutional Neural Networks:

- **Spatial Hierarchy in Data:** CNNs are excellent at capturing spatial hierarchies in data, which is crucial for image-related tasks. They can learn to detect edges in lower layers and complex patterns in higher layers.
- **Parameter Sharing:** Convolutional layers share weights across spatial locations, significantly reducing the number of parameters compared to fully connected layers, leading to more efficient models.
- **Translation Invariance:** Through pooling layers and the nature of convolution, CNNs are relatively invariant to translations, making them robust to shifts in the input image.
- **Automated Feature Extraction:** CNNs automatically learn to extract features from raw pixel data, eliminating the need for manual feature engineering.
- **Versatility:** Besides image recognition, CNNs have been adapted for various tasks, including object detection, image segmentation, and even speech and text processing.
- Disadvantages of Convolutional Neural Networks
- **High Computational Cost:** Training CNNs requires significant computational power, particularly for deep networks with many layers. This often necessitates the use of GPUs.
- **Large Amounts of Data Needed:** CNNs typically require large datasets to generalize well and avoid overfitting, which can be a limitation in domains with limited labeled data.
- **Complexity and Interpretability:** The deep and complex architecture of CNNs makes them harder to interpret compared to simpler models. Understanding why a CNN made a particular decision can be challenging.
- **Susceptibility to Adversarial Attacks:** CNNs can be vulnerable to adversarial attacks, where small perturbations

to the input data can lead to incorrect classifications with high confidence.

- **Tuning Hyperparameters:** CNNs have numerous hyperparameters (e.g., number of layers, filter sizes, stride, etc.) that require careful tuning, which can be time-consuming and computationally intensive.

4.4.3 Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNNs) are a class of neural networks designed to handle sequential data by maintaining a form of memory through their recurrent connections. Unlike feedforward neural networks, RNNs have connections that loop back, allowing information to persist and making them well-suited for tasks where the order of inputs is significant, such as time-series prediction, language modeling, and speech recognition.

Structure of Recurrent Neural Networks

Input Layer: Receives the sequential data. Each time step of the sequence is processed one at a time.

Hidden Layer(s): Each neuron in the hidden layer not only receives input from the current time step but also from the previous hidden state, enabling the network to retain information over time.

Output Layer: Produces the output for each time step or a single output after processing the entire sequence.

Example of Recurrent Neural Network

Consider an RNN designed to perform sentiment analysis on a sequence of words in a sentence.

Input Layer: The input is a sequence of words, e.g., "I love this movie."

Embedding Layer: Converts each word into a dense vector representation.

Recurrent Layer: Processes each word in the sequence one at a time, maintaining a hidden state that captures the context.

Fully Connected Layer: Maps the final hidden state to an output layer that predicts the sentiment (positive or negative).

Advantages of Recurrent Neural Networks

- **Handling Sequential Data:**RNNs are designed to handle sequential data, making them ideal for tasks where the order of inputs matters, such as time-series analysis, language modeling, and speech recognition.
- **Memory of Previous Inputs:**RNNs can retain information from previous inputs through their recurrent connections, enabling them to capture temporal dependencies and context.
- **Parameter Sharing:**Parameters are shared across different time steps, reducing the number of parameters and making the network more efficient.
- **Versatility:**RNNs can be used for various applications, including classification, regression, and generative tasks.

Disadvantages of Recurrent Neural Networks

- **Difficulty in Training:** RNNs can be challenging to train due to issues like vanishing and exploding gradients, which make it hard for the network to learn long-range dependencies.
- **Computationally Intensive:** Training RNNs can be computationally intensive, especially for long sequences, as the computation is sequential and cannot be easily parallelized.
- **Limited Long-Term Memory:** Basic RNNs struggle with retaining long-term dependencies. Advanced variants like LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) are often used to address this issue.
- **Complexity in Implementation:** Implementing RNNs and tuning their hyperparameters can be more complex compared to feedforward networks.

4.4.4 Long Short-Term Memory (LSTM) Network

A Long Short-Term Memory (LSTM) network is a type of artificial neural network designed to process sequences of data, such as time-series data or sentences, where the order of the data points matters.

LSTMs are especially good at remembering information for long periods, which helps them handle tasks where the context of previous data points is important.

How LSTMs Work

Imagine you're reading a story. To understand what's happening in the current sentence, you need to remember what happened in the previous sentences. An LSTM works similarly by maintaining a "memory" of previous inputs while processing new ones.

Here's a simple breakdown:

Memory Cell: This is where the LSTM stores information over time. It's like the memory of the network.

Gates: Gates control how information flows in and out of the memory cell. They decide what information to keep, what to add, and what to forget.

Forget Gate: Decides what information from the memory cell to forget.

Input Gate: Determines what new information to add to the memory.

Output Gate: Controls what information to output from the memory cell.

These gates use mathematical operations to make their decisions, based on the input data and the information already in memory.

Example: Sentiment Analysis

Let's say we want to use an LSTM to analyze the sentiment (positive or negative) of movie reviews.

Input: The LSTM receives a sequence of words from a movie review, one word at a time.

For example, the review might be: "I loved this movie."

Memory Cell and Gates: As each word is processed, the LSTM decides what to remember and what to forget.It keeps track of important words like "loved" which indicate a positive sentiment.

Output: After processing all the words, the LSTM outputs a sentiment score (e.g., a probability between 0 and 1, where closer to 1 means positive sentiment).

Advantages of LSTMs

- **Long-Term Memory:**LSTMs can remember information for long periods, making them effective for tasks where context over time is crucial.
- **Handling Sequences:**They are excellent for processing sequential data, like text or time-series data.
- **Flexibility:**LSTMs can be used for various applications, including language translation, speech recognition, and stock price prediction.

Disadvantages of LSTMs

Complexity:LSTMs are more complex and computationally intensive than simpler models.

Training Time: Training LSTMs can be slow, especially on large datasets.

Hyperparameter Tuning: They require careful tuning of hyperparameters, such as the number of units, learning rate, and batch size.

4.4.5 Gated Recurrent Unit (GRU) Network

A Gated Recurrent Unit (GRU) network is a type of recurrent neural network (RNN) designed to process sequential data, just like LSTMs. It's a variation of the traditional RNN architecture, aiming to address some of the limitations of standard RNNs, such as difficulties in learning long-term dependencies and vanishing gradients.

How GRUs Work

Think of a GRU as a simplified version of an LSTM. It also has mechanisms to retain memory over time, but it uses fewer gates and has a simpler structure compared to LSTMs.

Here’s a simplified breakdown:

Update Gate: This gate decides how much of the past information to keep and how much of the new information to consider.

Reset Gate: This gate decides how much of the past information to forget when considering new information.

Example: Language Modelling: Imagine you're predicting the next word in a sentence. A GRU can help by considering the words you've seen so far and predicting the next word based on that context.

Key Features of GRUs

Simplicity: GRUs have a simpler architecture compared to LSTMs, making them easier to understand and train.

Efficiency: Due to their simpler structure, GRUs are computationally less expensive than LSTMs, making them faster to train and execute.

Learning Long-Term Dependencies: While not as effective as LSTMs in learning very long-term dependencies, GRUs can still capture meaningful context over moderately long sequences.

4.4.6 Autoencoders

An autoencoder is a type of artificial neural network used for unsupervised learning of efficient data representations. It learns to encode input data into a lower-dimensional latent space representation and then decode it back to reconstruct the original input as accurately as possible. The architecture consists of an encoder network that compresses the input data into a latent representation and a decoder network that reconstructs the input from the latent representation.

Key Components of an Autoencoder:

Encoder: The encoder network compresses the input data into a lower-dimensional latent representation. It consists of multiple layers of neurons that progressively reduce the dimensionality of the input.

Decoder: The decoder network reconstructs the original input data from the compressed latent representation. It mirrors the architecture of the encoder, but in reverse, progressively expanding the dimensionality back to the original input shape.

Latent Space: The latent space is the lower-dimensional representation of the input data learned by the encoder. It captures the most important features of the input data while discarding unnecessary details.

How Autoencoders Work:

Encoding: The input data is fed into the encoder network, which compresses it into a lower-dimensional latent representation.

Decoding: The latent representation is then fed into the decoder network, which reconstructs the original input data from the latent representation.

Training: During training, the autoencoder learns to minimize the reconstruction error, typically measured using a loss function like mean squared error or binary cross-entropy. The weights of the encoder and decoder networks are adjusted iteratively using back propagation to minimize this error.

Applications of Autoencoders:

Dimensionality Reduction: Autoencoders can be used for reducing the dimensionality of data, which is useful for visualization, feature extraction, and data compression.

Anomaly Detection: They can also be used for anomaly detection by reconstructing normal data accurately and identifying deviations as anomalies.

Data Denoising: Autoencoders can remove noise from data by learning to reconstruct the clean data from noisy inputs.

Feature Learning: They can learn useful features from unlabeled data, which can then be used for supervised learning tasks.

Generative Modelling: Variants like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) can generate new data samples similar to the training data.

4.4.7 Generative Adversarial Network

A Generative Adversarial Network (GAN) is a type of deep learning model composed of two neural networks, the generator and the discriminator, which are trained simultaneously through a competitive process. GANs are used for generating new data samples that resemble a given dataset. The generator learns to produce realistic data samples, while the discriminator learns to distinguish between real data samples from the dataset and fake data samples generated by the generator.

Key Components of a GAN:

Generator: The generator takes random noise as input and generates synthetic data samples. It learns to map the noise distribution to the data distribution of the training dataset, producing realistic-looking samples.

Discriminator: The discriminator is a binary classifier that learns to distinguish between real data samples from the dataset and fake data samples generated by the generator. It is trained to assign high probabilities to real data samples and low probabilities to fake ones.

How GANs Work:

Training Process: The generator and discriminator are trained simultaneously in a minimax game framework. The generator aims to fool the discriminator by generating realistic samples that are indistinguishable from real ones, while the discriminator aims to correctly classify real and fake samples.

Generator Training: The generator takes random noise as input and generates synthetic samples. These samples are fed into the discriminator, which assigns probabilities to them. The generator is trained to maximize the probability of the discriminator making a mistake (i.e., classifying fake samples as real).

Discriminator Training: The discriminator is trained using real data samples from the dataset and fake samples generated by the generator. It learns to correctly classify real and fake samples by minimizing a loss function that penalizes misclassifications.

Adversarial Training: The generator and discriminator are trained iteratively in a competitive process. As the discriminator improves at distinguishing between real and fake samples, the generator learns to produce more realistic samples to fool the discriminator.

Applications of GANs:

Image Generation: GANs can generate realistic images of faces, animals, landscapes, and other objects.

Image-to-Image Translation: They can translate images from one domain to another, such as converting sketches to photos or day to night.

Super-Resolution: GANs can enhance the resolution of low-resolution images to produce high-resolution versions.

Style Transfer: They can transfer the style of one image onto another image while preserving its content.

Data Augmentation: GANs can generate synthetic data samples to augment small datasets for training machine learning models.

Anomaly Detection: GANs can detect anomalies in data by generating normal data samples and identifying deviations as anomalies.

Challenges of GANs:

Mode Collapse: GANs can suffer from mode collapse, where the generator learns to produce a limited set of samples, resulting in poor diversity.

Training Stability: Training GANs can be unstable, requiring careful hyperparameter tuning and regularization techniques to prevent divergence or oscillation.

Evaluation: Evaluating the performance of GANs can be challenging, as there is no direct metric to measure the quality of generated samples objectively.

4.4.8 Radial Basis Function Network

The Radial Basis Function (RBF) network is a type of artificial neural network that uses radial basis functions as activation functions. RBF networks are commonly used for function approximation, classification, and regression tasks. They are particularly well-suited for problems with smooth, nonlinear relationships between input and output variables.

Key Components of an RBF Network:

Input Layer: The input layer consists of neurons that represent the input variables. Each neuron receives an input value and passes it directly to the next layer.

Radial Basis Function Layer: The hidden layer of an RBF network consists of radial basis function neurons. Each neuron computes its output based on the distance between the input data and a center point, using a radial basis function as its activation function.

Output Layer: The output layer computes the final output of the network based on the weighted sum of the outputs from the radial basis function neurons in the hidden layer.

How RBF Networks Work:

Initialization: Initially, the centers and widths of the radial basis functions are randomly initialized or chosen based on some heuristic.

Training: The parameters of the RBF network, including the centers, widths, and weights, are optimized using a training algorithm such as gradient descent or the k-means algorithm.

Forward Propagation: During inference, input data is passed through the network. The radial basis function neurons in the hidden layer compute their outputs based on the distance between the input data and their respective center points.

Output Calculation: The outputs of the radial basis function neurons are combined in the output layer to produce the final output of the network.

Applications of RBF Networks:

Function Approximation: RBF networks are used to approximate complex, nonlinear functions in regression tasks.

Classification: They can be applied to classification problems where the decision boundaries are nonlinear.

Time-Series Prediction: RBF networks can predict future values in time-series data, such as stock prices or weather patterns.

Anomaly Detection: They are used for detecting anomalies or outliers in data by modeling the normal behavior of the system.

Control Systems: RBF networks are used in control systems for tasks such as adaptive control and system identification.

Advantages of RBF Networks:

Nonlinearity: RBF networks can model complex, nonlinear relationships between input and output variables.

Interpretability: The centers and widths of the radial basis functions provide insights into the regions of the input space where the network is most sensitive.

Scalability: RBF networks are relatively simple and can be trained efficiently, making them suitable for large-scale applications.

Disadvantages of RBF Networks:

Model Selection: Choosing appropriate values for the centers and widths of the radial basis functions can be challenging and may require domain knowledge or heuristic methods.

Overfitting: RBF networks are prone to overfitting, especially when the number of radial basis functions is too high relative to the size of the training dataset.

Limited Generalization: RBF networks may struggle to generalize to unseen data if the training data does not adequately represent the underlying distribution of the data.

4.4.9 Modular Neural Network

A Modular Neural Network (MNN) is a type of artificial neural network architecture composed of multiple interconnected modules, each designed to perform a specific sub-task or function. MNNs are characterized by their modular structure, which allows for the flexible composition and combination of different modules to solve complex problems.

Key Components of a Modular Neural Network:

Modules: Modules are individual neural network components that perform specific sub-tasks, such as feature extraction, classification, or memory storage. Each module can be designed independently and connected to other modules in a flexible manner.

Interconnections: Modules in an MNN are interconnected through a network topology that defines how information flows between them. The interconnections can be static or dynamic, depending on the requirements of the task.

Communication: Communication between modules is facilitated through message passing or data exchange mechanisms. Modules may send and receive information to coordinate their activities and share intermediate results.

How Modular Neural Networks Work:

Module Design: Each module in an MNN is designed to perform a specific function or sub-task. Modules can vary in complexity and can be implemented using different neural network architectures, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), or fully connected networks.

Interconnection: Modules are connected to each other based on the requirements of the task. The network topology defines the flow of information between modules, determining how data is processed and transformed as it propagates through the network.

Training: Modules in an MNN can be trained independently or jointly, depending on the training strategy and the level of interaction between modules. Training may involve optimizing the parameters of individual modules using backpropagation or reinforcement learning techniques.

Integration: Once trained, the modules are integrated into a cohesive network architecture. The integrated MNN can then be used to perform the desired task, such as image classification, natural language processing, or robotic control.

Applications of Modular Neural Networks:

Multimodal Learning: MNNs can integrate information from multiple modalities, such as text, images, and audio, to perform tasks that require multimodal input processing, such as multimedia content analysis or human-computer interaction.

Hierarchical Learning: MNNs can learn hierarchical representations of data by composing modules at different levels of abstraction. This allows for the extraction of increasingly complex features from raw input data, enabling tasks such as object recognition or speech understanding.

Transfer Learning: Modular neural networks can facilitate transfer learning by reusing pre-trained modules across different tasks or domains. This can speed up the training process and improve the performance of the network on new tasks with limited labeled data.

Adaptive Systems: MNNs can be used to build adaptive systems that can dynamically reconfigure their architecture or adjust their behavior based on changing environmental conditions or task requirements. This makes them suitable for applications such as autonomous robotics or intelligent decision-making.

Advantages of Modular Neural Networks:

Flexibility: MNNs offer flexibility in network design and composition, allowing for the integration of diverse modules tailored to specific tasks or domains.

Scalability: Modular architectures can scale to handle complex problems by decomposing them into smaller, more manageable sub-problems that can be solved independently.

Interpretability: The modular structure of MNNs facilitates interpretability by allowing researchers to analyze the contributions of individual modules to the overall network performance.

Challenges of Modular Neural Networks:

Module Integration: Integrating modules into a cohesive network architecture can be challenging, requiring careful design choices and optimization to ensure compatibility and interoperability between modules.

Training Complexity: Training MNNs can be complex, especially when modules interact with each other or share parameters. Coordination between modules during training may require additional mechanisms to prevent interference or instability.

Computational Cost: The modular structure of MNNs may introduce additional computational overhead, particularly when modules need to communicate or exchange information frequently during inference or training.

4.4.10 Spiking Neural Network

A Spiking Neural Network (SNN) is a type of artificial neural network inspired by the biological neural networks found in the brain. Unlike traditional artificial neural networks, which use continuous-valued signals (e.g., activations), SNNs communicate through discrete spikes or pulses of activity, mimicking the behavior of neurons in the brain.

Key Components of a Spiking Neural Network:

Neurons: Neurons in an SNN model the behavior of biological neurons. Each neuron receives input spikes from other neurons, integrates them over time, and generates output spikes based on its internal state and activation function.

Synapses: Synapses are connections between neurons through which spikes are transmitted. Each synapse has a weight that determines the strength of the connection between neurons and modulates the impact of incoming spikes on the receiving neuron.

Spiking Dynamics: In SNNs, neurons generate output spikes based on their membrane potential, which is influenced by the incoming spikes and decays over time. When the membrane potential exceeds a certain threshold, the neuron emits a spike and resets its membrane potential.

How Spiking Neural Networks Work:

Encoding Input: Input data is encoded into spike trains, where each input feature is represented by a series of spikes over time. Different encoding schemes can be used, such as rate coding or temporal coding.

Propagation: Spikes propagate through the network from the input layer to the output layer, passing through intermediate layers of neurons. At each neuron, incoming spikes are integrated over time, affecting the membrane potential and determining the timing of output spikes.

Learning: Synaptic weights in the network are adapted through learning algorithms to optimize network performance. Spike-timing-dependent plasticity (STDP) is a common learning rule used in SNNs, which strengthens or weakens synapses based on the relative timing of pre- and post-synaptic spikes.

Applications of Spiking Neural Networks:

Neuromorphic Computing: SNNs are used in neuromorphic computing systems, which aim to emulate the brain's computing principles for energy-efficient and brain-inspired computing architectures.

Temporal Processing: SNNs are well-suited for tasks that involve processing temporal information, such as speech recognition, time-series prediction, and event-based vision.

Event-Based Sensors: SNNs can be applied to process data from event-based sensors, such as event-based cameras or event-based auditory sensors, which generate spikes in response to changes in the environment.

Bio-Inspired Robotics: SNNs are used in bio-inspired robotics for controlling robots and autonomous agents, enabling them to exhibit more adaptive and lifelike behavior.

Advantages of Spiking Neural Networks:

Biological Plausibility: SNNs closely mimic the spiking behavior of biological neurons, making them a more biologically plausible model of neural computation.

Temporal Dynamics: SNNs naturally capture the temporal dynamics of data, allowing them to process information over time and encode temporal patterns more effectively than traditional neural networks.

Energy Efficiency: The event-driven nature of SNNs enables energy-efficient computation, as spikes are only generated when necessary, reducing overall power consumption.

Challenges of Spiking Neural Networks:

Learning Complexity: Learning in SNNs can be more challenging than in traditional neural networks due to the discrete nature of spike-based communication and the complexity of spike-timing-dependent plasticity rules.

Hardware Implementation: Implementing SNNs in hardware requires specialized neuromorphic hardware that can efficiently simulate the spiking behavior of neurons and synapses.

Sparse Representations: SNNs often operate with sparse representations, which can require additional computational resources for efficient processing and storage.

4.5 Neural Network for Clustering

Neural networks for clustering are a type of artificial neural network (ANN) designed specifically for unsupervised learning tasks, particularly clustering. Clustering is the process of grouping similar data points together into clusters or categories based on their intrinsic characteristics. Neural networks for clustering leverage the power of neural networks to automatically discover patterns and structure in data without the need for labeled training data.

Key Components of Neural Networks for Clustering:

Input Layer: The input layer of the neural network receives the input data, which consists of features representing the characteristics of each data point.

Hidden Layers: The hidden layers of the neural network perform the main computational processing. Each neuron in the hidden layers aggregates information from the input data and passes it through an activation function to produce an output.

Output Layer: The output layer of the neural network produces the final output, which represents the cluster assignments for each data point. The output layer may consist of one neuron per cluster, with each neuron representing the likelihood or membership of a data point belonging to a particular cluster.

How Neural Networks for Clustering Work:

Initialization: The neural network is initialized with random weights and biases.

Forward Propagation: Input data is fed into the neural network, and forward propagation is performed to compute the output of each neuron in the hidden layers and the output layer.

Cluster Assignment: The output of the neural network represents the cluster assignments for each data point. Data points are assigned to the cluster with the highest output value.

Training: The neural network is trained using an unsupervised learning algorithm, such as the k-means algorithm or self-organizing maps (SOMs). During training, the weights and biases of the neural network are adjusted to minimize an objective function, such as the within-cluster sum of squares or the quantization error.

Convergence: Training continues until the neural network converges to a stable solution, where the cluster assignments no longer change significantly with each iteration.

Applications of Neural Networks for Clustering:

Pattern Recognition: Neural networks for clustering are used for pattern recognition tasks, such as image segmentation, where pixels with similar characteristics are grouped together into regions or objects.

Customer Segmentation: In marketing and customer analytics, neural networks for clustering are used to segment customers into groups based on their purchasing behavior, demographics, or preferences.

Anomaly Detection: Neural networks for clustering can be used for anomaly detection by identifying data points that do not belong to any of the clusters, indicating potential outliers or anomalies.

Document Clustering: In natural language processing, neural networks for clustering are used to cluster documents based on their content, enabling tasks such as document categorization or topic modeling.

Advantages of Neural Networks for Clustering:

Flexibility: Neural networks for clustering are flexible and can adapt to complex data distributions and non-linear relationships between features.

Scalability: They can handle large-scale datasets with high-dimensional feature spaces, making them suitable for real-world applications with diverse data sources.

Automation: Neural networks for clustering automate the process of discovering patterns and structure in data, reducing the need for manual intervention and domain expertise.

Disadvantages of Neural Networks for Clustering:

Complexity: Neural networks for clustering can be complex to train and interpret, requiring expertise in neural network architecture design and optimization techniques.

Computationally Intensive: Training neural networks for clustering can be computationally intensive, particularly for large-scale datasets with many features and clusters.

Overfitting: Like other machine learning models, neural networks for clustering are susceptible to overfitting, where the model learns to memorize the training data rather than generalize to unseen data.

4.6 Neural Network for Feature Extraction

Neural networks for feature extraction are artificial neural networks designed to automatically learn and extract meaningful representations or features from raw input data. These learned features capture important characteristics of the data that are useful for downstream tasks such as classification, regression, or clustering. Feature extraction is a crucial step in many machine learning and pattern recognition applications, as it helps reduce the dimensionality of the data and highlight relevant information for subsequent analysis.

Key Components of Neural Networks for Feature Extraction:

Input Layer: The input layer of the neural network receives the raw input data, which can be images, text, audio, or any other type of structured or unstructured data.

Hidden Layers: The hidden layers of the neural network perform the main computational processing and feature extraction. Each neuron in the hidden layers aggregates information from the input data and passes it through an activation function to produce an output.

Output Layer: The output layer of the neural network produces the final output, which represents the learned features or representations of the input data. The output layer may consist of one or more neurons, depending on the dimensionality and complexity of the feature space.

How Neural Networks for Feature Extraction Work:

Initialization: The neural network is initialized with random weights and biases.

Forward Propagation: Input data is fed into the neural network, and forward propagation is performed to compute the output of each neuron in the hidden layers and the output layer.

Feature Learning: During training, the neural network learns to extract meaningful features from the input data by adjusting the weights and biases of the network using a supervised or unsupervised learning algorithm. The learned features are optimized to minimize a loss function that measures the discrepancy between the predicted and actual outputs.

Fine-Tuning: After feature extraction, the learned features can be further refined or fine-tuned using additional training data or optimization techniques to improve their discriminative power or generalization performance.

Applications of Neural Networks for Feature Extraction:

Computer Vision: In computer vision tasks such as object detection, image classification, and image segmentation, neural networks for feature extraction are used to extract hierarchical features from raw pixel data, enabling the detection and recognition of objects and patterns in images.

Natural Language Processing: In natural language processing tasks such as sentiment analysis, named entity recognition, and machine translation, neural networks for feature extraction are used to learn distributed representations of words or phrases from text data, capturing semantic and syntactic relationships between words.

Speech Recognition: In speech recognition tasks, neural networks for feature extraction are used to extract spectrotemporal features from audio signals, such as mel-frequency cepstral coefficients (MFCCs) or spectrograms, which are then fed into a speech recognition system to transcribe spoken language into text.

Healthcare: In healthcare applications such as medical imaging analysis and disease diagnosis, neural networks for feature extraction are used to extract informative features from medical images or patient data, facilitating the detection and diagnosis of diseases and abnormalities.

Advantages of Neural Networks for Feature Extraction:

Automatic Feature Learning: Neural networks for feature extraction automatically learn to extract relevant features from raw input data, eliminating the need for manual feature engineering and domain expertise.

Hierarchical Representations: These networks can learn hierarchical representations of data, capturing complex and abstract relationships between features at multiple levels of abstraction.

Adaptability: Neural networks for feature extraction are highly adaptable and can be tailored to specific tasks or domains by adjusting the network architecture, hyper parameters, and training objectives.

Disadvantages of Neural Networks for Feature Extraction:

Computational Complexity: Training neural networks for feature extraction can be computationally intensive, particularly for large-scale datasets or complex network architectures.

Overfitting: Like other machine learning models, neural networks for feature extraction are susceptible to overfitting, where the model learns to memorize the training data rather than generalize to unseen data. Regularization techniques and data augmentation strategies can help mitigate overfitting.

Interpretability: The learned features extracted by neural networks may lack interpretability, making it difficult to understand and interpret the underlying representations learned by the model.

CHECK YOUR PROGRESS

A. Multiple Choice Questions (MCQs) with Answers

1. What is a Feedforward Neural Network (FNN)?

(a) A neural network where connections between nodes form cycles

(b) A neural network primarily used for sequential data

(c) A neural network where connections between nodes do not form cycles

(d) A neural network that uses radial basis functions as activation functions

2. Which type of neural network is commonly used for image processing?

(a) RNN

(b) CNN

(c) GAN

(d) Autoencoder

3. What is the primary purpose of an autoencoder?

(a) To classify data into categories

(b) To learn efficient codings of input data

(c) To predict future values in a sequence

(d) To generate realistic data samples

4. Which neural network variant is designed to handle long-term dependencies?

(a) CNN

(b) GAN

(c) RBF Network

(d) LSTM

5. What does a Generative Adversarial Network (GAN) consist of?

(a) A generator and a discriminator

(b) An encoder and a decoder

(c) A cluster and a classifier

(d) A spike and a neuron

6. Which activation function is commonly used to introduce non-linearity into a neural network?

(a) Linear

(b) ReLU

(c) Polynomial

(d) Exponential

7. What technique is used to prevent overfitting in neural networks?

(a) Regularization

(b) Linear regression

(c) Clustering

(d) Forward propagation

8. In an autoencoder, what is the purpose of the bottleneck layer?

(a) To expand the data dimensions

(b) To capture the most important features

(c) To initialize weights and biases

(d) To serve as the output layer

9. What is the key advantage of using Recurrent Neural Networks (RNNs)?

(a) Handling high-dimensional data

(b) Learning temporal dependencies in sequences

(c) Reducing computational complexity

(d) Providing interpretable results

10. Which type of neural network uses spikes or discrete events to process information?

(a) CNN

(b) GAN

(c) SNN

(d) RBF

B. State whether the following statements are true or false:

1. Convolutional Neural Networks (CNNs) are primarily used for processing sequential data.

2. Long Short-Term Memory (LSTM) networks can handle long-term dependencies in sequential data.

3. Gated Recurrent Unit (GRU) networks have more gates than LSTM networks.

4. Autoencoders are used for generating realistic data samples.

5. Generative Adversarial Networks (GANs) consist of an encoder and a decoder.

6. Radial Basis Function (RBF) networks use radial basis functions as activation functions in their hidden layers.

7. Modular Neural Networks (MNNs) are composed of multiple smaller networks that work independently on sub-tasks.

8. Spiking Neural Networks (SNNs) model neural activity using spikes or discrete events.

9. Neural networks can be used for both feature extraction and clustering tasks.

10. Regularization techniques are used to encourage overfitting in neural networks.

4.7 Summing Up

1. In this chapter, we explored the multifaceted world of neural networks, focusing on their applications in feature extraction and clustering. We began by introducing neural networks and their importance in modern machine learning, emphasizing their ability to model complex relationships and automatically learn features from raw data.

2. We then detailed various types of neural networks, including Feedforward Neural Networks (FNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) networks, Gated Recurrent Unit (GRU) networks, Autoencoders, Generative Adversarial Networks (GANs), Radial Basis Function (RBF) networks, Modular Neural Networks (MNN), and Spiking Neural Networks (SNN). Each type was described with its working mechanism, example applications, and the respective advantages and disadvantages.

3. Special attention was given to CNNs and RNNs due to their prominent roles in feature extraction and handling sequential data, respectively. We illustrated the feature extraction process using these networks and highlighted how they automatically identify relevant features from input data, which is crucial for tasks such as image and speech recognition.

4. We also covered neural networks for clustering, explaining how they group similar data points without predefined labels and discussing their key components, applications, advantages, and disadvantages.

5. Throughout the chapter, practical examples were provided to solidify understanding, and the challenges associated with neural networks, such as computational complexity, data requirements, and interpretability, were discussed. This comprehensive exploration equips readers with a deep understanding of how neural networks can be leveraged for feature extraction and clustering, preparing them to apply these powerful tools in various data-driven scenarios.

4.8 Answers to Check Your Progress

A 1. c 2.b 3.b 4.d 5.a 6.b 7.a 8.b 9.b 10.c

B. 1. False	2. True	3.False	4. False
5. False	6. True	7. True	8. True
9. True	10. False		

4.9 Possible Questions

1. Explain the concept of overfitting in neural networks.
2. What are the main differences between LSTM and GRU networks?
3. Describe the role of an activation function in a neural network.
4. What is feature extraction, and why is it important in machine learning?
5. How do convolutional layers in a CNN help in feature extraction from images?
6. Define clustering and explain how neural networks can be used for this task.
7. What are the advantages of using neural networks for feature extraction?
8. Describe the architecture of an autoencoder.
9. Explain how Generative Adversarial Networks (GANs) are trained.
10. What is the purpose of regularization in neural networks?
11. Discuss the architecture and working mechanism of a Convolutional Neural Network (CNN). Provide a simple example of its application.
12. Explain the process of feature extraction using neural networks. Include differenttypes of neural networks used for this purpose and their applications.

13. Describe the advantages and disadvantages of using neural networks for clustering.
14. Provide a detailed explanation of Generative Adversarial Networks (GANs),including their architecture, training process, and applications.
15. What are Radial Basis Function (RBF) Networks, and how are they used in machinelearning? Explain their working mechanism and provide an example application.

4.10 References and Suggested Readings

[1] Gurney, K. (2018). *An introduction to neural networks*. CRC press.

[2] Khanna, T. (1990). *Foundations of neural networks*. Addison-Wesley Longman Publishing Co., Inc..

[3] Chester, M. (1993). *Neural networks: a tutorial*. Prentice-Hall, Inc..

[4] Freeman, J. A., & Skapura, D. M. (1991). *Neural networks: algorithms, applications, and programming techniques*. Addison Wesley Longman Publishing Co., Inc.

---×---

UNIT: 5
ROUGH SETS AND THEIR APPLICATION

Unit Structure:

5.1 Introduction

The notion of ***Rough sets*** was introduced by Z Pawlak in his seminal paper of 1982 (Pawlak 1982). It is a formal theory derived from fundamental research on logical properties of information systems. Rough set theory has been a methodology of database mining or knowledge discovery in relational databases. In its abstract form, it is a new area of uncertainty mathematics closely related to fuzzy theory. We can use rough set approach to discover structural relationship within imprecise and noisy data. Rough sets and fuzzy sets are complementary generalizations of classical sets. The approximation spaces of rough set theory are sets with multiple memberships, while fuzzy sets are concerned with partial memberships. The rapid development of these two approaches provides a basis for “soft computing,” initiated by Lotfi A. Zadeh. Soft Computing includes along with rough sets, at least fuzzy logic, neural networks, probabilistic reasoning, belief networks, machine learning, evolutionary computing, and chaos theory.

Because of the development of computer science and technology, especially the development of computer network, a large amount of information is provided for people every second of the day. With the growing amount of information, the requirement for information analysis tools is also becoming higher and higher, and people hope to automatically acquire the potentialknowledge from the data. Especially in the past 30 years, knowledge discovery (rule extraction, data mining, machine learning, etc.) has attracted much attention in the field of artificial intelligence. Thus, various kinds of knowledge discovery methods came into being.

Proposed by Professor Pawlak in 1982, the rough set theory is an important mathematical tool to deal with imprecise, inconsistent, incomplete information and knowledge. Originated from the simple information model, the basic idea of the rough set theory can be divided into two parts. The first part is to form concepts and rules through the classification of relational database. And the second part is to discovery knowledge through the classification of the equivalence relation and classification for the approximation of the target. As a theory of data analysis and processing, the rough set theory is a new mathematical tool to deal with uncertain information after probability theory, fuzzy set theory, and evidence theory. For the fuzzy set theory, membership function is a key factor. However, the selection of membership function is uncertain. Therefore, in a sense, the fuzzy set theory is an uncertain mathematical tool to solve the uncertain problems. In rough set theory, two precise boundary lines are established to describe the imprecise concepts. Therefore, in a sense, the rough set theory is a certain mathematical tool to solve the uncertain problems.

5.2 Unit Objective

By the end of this unit, students will be able to:

- Explain the fundamental concepts and principles of rough sets, including lower and upper approximations.
- Apply rough set theory to classify and analyze data with uncertainty and imprecision.
- Implement rough set-based methods for feature selection and rule generation in machine learning and data mining tasks.
- Evaluate the effectiveness of rough sets in handling noisy and incomplete datasets.
- Explore practical applications of rough sets in various domains, such as artificial intelligence and decision support systems.

5.3 Basic Concepts of Rough Set

A rough set is a mathematical concept used in data analysis to deal with vagueness and uncertainty. It was introduced by Zdzisław Pawlak in the early 1980s. Rough set theory focuses on the approximation of sets by employing a pair of sets called the lower and upper approximations. These approximations are used to handle and reason about uncertain or imprecise information.

Key Concepts of Rough Sets

- ➢ **Universe of Discourse (U)**: The set of all objects under consideration.
- ➢ **Indiscernibility Relation (R)**: An equivalence relation that groups objects having indistinguishable or similar attributes. If objects xx and yy are indiscernible, they belong to the same equivalence class.
- ➢ **Lower Approximation (L) of a Set**: The set of all objects that definitely belong to the target set based on the available information.
- ➢ **Upper Approximation (U) of a Set**: The set of all objects that possibly belong to the target set based on the available information.
- ➢ **Boundary Region**: The difference between the upper and lower approximations. It consists of objects that cannot be

classified with certainty as either belonging or not belonging to the target set.

Example

Consider a set of students and their scores in a mathematics test. We want to classify students into those who "Pass" and those who "Fail" based on their scores. Let's define the universe of discourse *U* and the attributes for the students.

Universe (U): {Anita, Bharat, Chitra, Deepak, Esha}

Attribute: Score

Assume the following scores for the students:

Anita: 85
Bharat: 90
Chitra: 70
Deepak: 75
Esha: 60

We set the threshold for passing as 75. Students with scores 75 or above pass; otherwise, they fail.

Now, define the indiscernibility relation *R* based on whether students have the same grade range. For simplicity, we consider two ranges: "Pass" (scores 75 and above) and "Fail" (scores below 75).

Equivalence Classes:

{Anita, Bharat} (both have scores 75 and above)
{Chitra, Deepak} (scores between 70 and 75)
{Esha} (score below 70)

Let's identify the rough set approximations for the "Pass" set.

- **Lower Approximation (L) of Pass**: The set of students who definitely pass based on their scores.
  L(Pass) = {Anita, Bharat}
- **Upper Approximation (U) of Pass**: The set of students who possibly pass based on their scores.
  U(Pass) = {Anita, Bharat, Chitra, Deepak}
- **Boundary Region**: The set of students who cannot be classified with certainty as either passing or failing.
  Boundary(Pass) = U(Pass) - L(Pass) = {Chitra, Deepak}

In this example:

- **Lower Approximation**: Anita and Bharat definitely pass (as their scores are clearly above the threshold).
- **Upper Approximation**: Anita, Bharat, Chitra, and Deepak possibly pass (as Chitra and Deepak have scores around the threshold).
- **Boundary Region**: Chitra and Deepak are in the boundary region because we cannot determine with certainty if they pass based on the given indiscernibility relation.
- **Not in Upper Approximation**: Esha is not included in the upper approximation since she definitely fails based on her score.

5.4 Mathematical Definition of Rough SET

A rough set is a mathematical framework for handling uncertainty and vagueness in data. It involves approximating a set Xwithin a universe Uusing two other sets: the lower approximation (X), which includes all elements that definitely belong to X, and the upper approximation (X), which includes all elements that possibly belong to X. Formally, the lower approximation (X)=$\{x \in U \mid [x]_R \subseteq X\}$ consists of all elements whose equivalence class under an indiscernibility relation $R$is entirely contained in X, and the upper approximation $U(X)$=$\{x \in U \mid [x]_R \cap X \neq \emptyset\}$ consists of all elements whose equivalence class intersects with X. The boundary region, (X)=$U(X)-L(X)$, includes elements that cannot be definitively classified as either belonging to Xor not.

Example: Let's consider a simple example with a small universe and a single attribute.

Universe (U): {A1, A2, A3, A4, A5}

Attribute: Score

Assume the following scores for the objects:

A1: 85
A2: 90
A3: 70
A4: 75
A5: 60
Let the set Xrepresent students who "Pass" with a threshold score of 75.

Pass (X): {A1, A2, A4}

Now, define the indiscernibility relation R based on similar scores:

Equivalence Classes:
{A1, A2} (both have scores above 75)
{A3, A4} (scores around 75)
{A5} (score below 75)

Using these equivalence classes, we can find the lower and upper approximations of the set X:

Lower Approximation (L(X)): {A1, A2}

A1 and A2 definitely belong to the "Pass" set.

Upper Approximation (U(X)): {A1, A2, A3, A4}

A1, A2, A3, and A4 possibly belong to the "Pass" set.

Boundary Region (B(X)): {A3, A4}

A3 and A4 are in the boundary region as we cannot determine with certainty if they pass based on the given information.

5.5 Applications of Rough SET

Rough set theory has been widely applied in various fields due to its ability to handle vagueness and uncertainty in data. Here are some key applications of rough set theory:

1. Data Mining and Knowledge Discovery

Rough set theory is extensively used in data mining for:

Feature Selection and Reduction: Identifying and selecting the most significant attributes from large datasets while reducing redundancy. This helps in simplifying models and improving their interpretability.

Classification and Clustering: Rough sets are used to create classification models and to group similar objects into clusters. They provide a way to handle uncertain and imprecise class boundaries.

Rule Generation: Deriving decision rules from data. Rough sets can extract if-then rules that are easily interpretable and useful for decision-making processes.

2. Decision Support Systems

In decision support systems, rough sets help in:

Multi-Criteria Decision Analysis (MCDA): Evaluating and prioritizing multiple conflicting criteria to support decision-making.

Risk Assessment: Analyzing uncertain and incomplete data to assess risks and make informed decisions.

3. Medical Diagnosis

Rough set theory is applied in medical fields for:

Disease Diagnosis and Prognosis: Analyzing patient data to diagnose diseases, predict disease progression, and recommend treatments.

Pattern Recognition: Identifying patterns in medical data to detect anomalies or specific conditions.

4. Image Processing and Pattern Recognition

Rough sets are used in image processing and pattern recognition for:

Image Segmentation: Dividing an image into meaningful regions based on rough set approximations.

Feature Extraction: Identifying important features in images that are essential for recognition tasks.

5. Machine Learning and Artificial Intelligence

In machine learning, rough sets contribute to:

Learning Algorithms: Developing algorithms that can learn from data with uncertainty and incomplete information.

Knowledge Representation: Representing and reasoning with knowledge that has imprecise and vague boundaries.

6. Economics and Financial Analysis

Rough set theory aids in:

Market Analysis: Analyzing market trends and consumer behavior under uncertainty.

Credit Scoring: Evaluating credit risk and determining creditworthiness based on incomplete financial data.

7. Engineering and Control Systems

In engineering, rough sets are used for:

Fault Diagnosis: Detecting and diagnosing faults in complex systems.

Quality Control: Monitoring and ensuring quality in manufacturing processes.

8. Linguistics and Natural Language Processing

Rough set theory is applied in linguistics for:

Text Classification: Categorizing text documents into predefined categories.

Semantic Analysis: Analyzing the meaning and relationships between words and phrases in text.

9. Environmental Science

In environmental science, rough sets help in:

Pollution Control: Analyzing data related to environmental pollution and predicting pollution levels.

Resource Management: Managing natural resources under uncertainty and incomplete data.

10. Social Science

Rough set theory assists in social sciences for:

Survey Analysis: Analyzing survey data with uncertain and imprecise responses.

Behavioral Studies: Studying human behavior patterns and social interactions.

5.6 Reduct and Rule Extraction in Rough SET Theory

Rough set theory helps handle uncertain and imprecise data by focusing on the most essential information. Two key concepts in this theory are reducts and rule extraction.

5.6.1.What is Reduct?

Reducts are the minimal sets of attributes that provide the same classification power as the entire set of attributes. This means finding the smallest number of characteristics needed to make the same decisions as if we used all characteristics.

Reducts are essential because they help in simplifying data without losing important information. In real-world applications, datasets often contain many attributes, some of which might be redundant or irrelevant. By identifying the reducts, we focus only on the most significant attributes, making the analysis more efficient and easier to interpret.

Example of Reducts

Consider a group of students described by three attributes: whether they **Studied (Yes/No)**, **Attended Classes (Yes/No)**, and **DidHomework (Yes/No)**. We want to predict if these students **Passed (Yes/No)**.

Here's a simple table:

Student	Studied	Attended Classes	Did Homework	Passed
A	Yes	Yes	Yes	Yes
B	Yes	No	Yes	No
C	No	Yes	Yes	No
D	Yes	Yes	No	Yes
E	no	No	Yes	No

To determine if we can predict "Passed" with fewer attributes, we look for a reduct. Suppose we find that just Studied and Did Homework are sufficient to classify students:

Student	Studied	Did Homework	Passed
A	Yes	Yes	Yes
B	Yes	Yes	No
C	No	Yes	No

D	Yes	No	Yes
E	no	Yes	No

This reduced table shows that "Attended Classes" is not necessary for classification. By using fewer attributes, we simplify our analysis while retaining the same predictive power.

5.6.2 Why Rule Extraction?

Rule extraction involves deriving simple if-then rules from data to make decisions or classifications. These rules are based on the relationships found in the reducts.

Example of Rule Extraction

Using the reduced table, we can extract rules that explain when a student passes:

Rule 1: If a student Studied = Yes and Did Homework = Yes, then Passed = Yes.

This rule is derived from Student A, who studied, did homework, and passed.

Rule 2**:** If a student Studied = Yes and Did Homework = No, then Passed = Yes.

This rule is derived from Student D, who studied, did not do homework, but still passed.

Rule 3: If a student Studied = No and Did Homework = Yes, then Passed = No.

This rule is derived from Students B, C, and E, who did not study (or did not study and did homework) and did not pass.

5.7 FUZZY SUBSETS

The notion of fuzzy set was introduced by L. Zadeh. Since then its application has been evident in different fields of study. This section introduces the basic definitions, notations and operations for fuzzy subsets of a non-empty set relevant for our study. Research on fuzzy subsets has been underway for over 50 years now. It is therefore impossible to cover all aspects in this field. We merely aim to provide a summary of the basic concepts central to the study of fuzzy subsets.

5.7.1 Definitions and Notations

Let E be a space of objects and x be a generic element of E. A classical set A, $A \subseteq E$ is defined as a collection of elements or objects $x \in E$, such that each element (x) can either belong or not to the set A. By defining a characteristic (or membership) function for each element x in E, we can represent a classical set A by a set of ordered pairs $(x,0)$ or $(x,1)$, which indicates $x \notin A$ or $x \in A$, respectively. Unlike the aforementioned conventional set, a fuzzy set expresses the degree to which an element belongs to a set. Hence the membership function of a fuzzy set is allowed to have values between 0 and 1, which denote the degree of membership of an element in the given set.

A fuzzy subset, A of E is characterized by a membership function $\mu_A : E \to [0,1] = I$ such that the number $\mu_A(x)$ in the unit interval I is interpreted as the degree of membership of element x to the fuzzy subset A for each $x \in E$. The set E is referred to as the universe of discourse.

Every $\{0,1\}$-valued fuzzy subset with membership function taking only either 0 or 1 is called a crisp subset, that is just a subset of E in the usual sense of the term. This means each object x of E either belongs to the subset A when the degree of membership is 1 or does not belong to the subset A when the membership is 0 of E.

It is mathematically defined as, $A = \{(x, \mu_A(x)) : x \in A, \mu_A(x) \in [0,1]\}$.

For example $E=\{x_0,x_1,x_2\}$ such that $\mu(x_0)=0.1$, $\mu(x_1)=0.7$, $\mu(x_2)=0.4$. Therefore, $A=\{(x_0,0.1),(x_1,0.7),\}(x_2,0.4)$ is a fuzzy subset of E.

The set such as $M=\{0.1,0.7,0.4\}$ in the above case is called the membership set of the fuzzy subset A.

Obviously, the definition of a fuzzy set is a simple extension of the definition of a classical (crisp) set in which the characteristic function is permitted to have any values between 0 and 1. If the values of the membership function are restricted to either 0 or 1, then A is restricted to a classical set. For clarity, we shall also refer to classical sets as ordinary sets, non-fuzzy sets, crisp sets or just sets. Usually E is referred to as the **Universe of discourse**, or simply we shall call the **Universal set**.

Therefore, the fuzzy subset containing membership value $'1'$ for all elements is called the **Universal fuzzy set**or **universal set** (**1**) and the fuzzy subset containing membership value $'0'$ for all elements is called the **Empty fuzzy set** (0) or empty set. Both these two sets are actually crisp sets.

5.7.2 Basic Operations

If A and B are two fuzzy subsets of a set $E=\{x_0,x_1,...x_{n-1}\}$ defined by-

$$A=\{(x,\mu_A(x)):x\in E,\mu_A(x)\in[0,1]\}.$$
$$B=\{(x,\mu_B(x)):x\in E,\mu_B(x)\in[0,1]\}.$$

Then some basic operations on A and B are defined as follows:-

Equality: The fuzzy subsets A and B are equal denoted by $A = B$ if and only if for every $x \in E$, $\mu_A = \mu_B$

Inclusion: The fuzzy set A is said to be included in the fuzzy set B denoted by $A \subseteq B$ if for every $x \in E,\ \mu_A(x) \leq \mu_B(x)$. A is called the subset of B.

Proper subset: The fuzzy subset A is called a proper subset of the fuzzy subset B denoted $A \subset B$ when A is a subset of B and $A \neq B$, that is

$\mu_A(x) \leq \mu_B(x)$ for every $x \in E$

$\mu_A(x) < \mu_B(x)$ for at least one $x \in E$

Intersection: The intersection of A and B is denoted by $A \cap B$ and this operation is mathematically defined as: $\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)), x \in E.$

Union: The union of A and B is denoted by $A \cup B$ and defined as: $\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x)), x \in E.$

Complementation: For any fuzzy set A, its complement is denoted by A' and defined as $\mu_{A'}(x) = 1 - \mu_A(x), x \in E$.

Difference: The difference between the two fuzzy sets A and B is denoted by $A - B$ and defined as: $\mu_{A-B}(x) = \min(\mu_A(x), \mu_{B'}(x)), x \in E.$

5.7.3 Properties of FUZZY subsets

If A, B and C are three fuzzy subsets of a set E, then they satisfy the following properties:

Commutative laws:

a) $A \cap B = B \cap A$ b) $A \cup B = B \cup A$

Distributive laws:

a) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ b)
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

Identity laws:

a) $A \cup 0 = A$ b) $A \cap 1 = A$

Idempotent laws:

a) $A \cup A = A$ b) $A \cap A = A$

Absorption laws:

a) $A \cup (A \cap B) = A$ b) $A \cap (A \cup B) = A$

Associative laws:

a) $(A \cup B) \cup C = A \cup (B \cup C)$ b) $(A \cap B) \cap C = A \cap (B \cap C)$

Involution law:

$(A')' = A$

DE Morgan's laws

a) $(A \cap B)' = A' \cup B'$ b) $(A \cup B)' = A' \cap B'$

5.8 Applications of FUZZY SET Theory

Fuzzy set theory, introduced by Lotfi Zadeh in 1965, provides a mathematical framework for dealing with uncertainty and imprecision, making it highly applicable in various areas of computer science. Here are some key applications:

- **Artificial Intelligence and Expert Systems**: Fuzzy logic is used to emulate human reasoning in AI systems, allowing for decision-making in environments with uncertainty and imprecise information.
- **Machine Learning and Data Mining**: Fuzzy sets help in handling imprecise data, enabling better classification, clustering, and pattern recognition. They improve algorithms by dealing with overlapping classes and ambiguous boundaries.
- **Control Systems**: Fuzzy controllers, such as those used in consumer electronics (e.g., washing machines, air conditioners), use fuzzy logic to handle varying inputs and maintain optimal performance.
- **Natural Language Processing**: Fuzzy set theory helps in managing the ambiguity and vagueness inherent in human language, improving tasks like text classification, sentiment analysis, and information retrieval.
- **Image Processing and Computer Vision**: Fuzzy techniques are applied to enhance, segment, and recognize patterns in images, accommodating the imprecise nature of visual data.
- **Decision Support Systems**: Fuzzy logic aids in making complex decisions by incorporating human-like reasoning in environments with uncertain and incomplete information.
- **Robotics**: Fuzzy set theory is used in robot navigation and control, allowing robots to make decisions based on uncertain sensor data and operate in dynamic, real-world environments.
- **Bioinformatics**: In analyzing biological data, fuzzy logic handles the inherent uncertainty and variability, improving gene expression analysis, protein structure prediction, and disease diagnosis.

Check Your Progress

1. What is the primary purpose of Rough Set Theory?

A) To manage imprecise and uncertain information

B) To provide statistical analysis tools

C) To perform data encryption

D) To create neural networks

2. Which component in Rough Set Theory is essential for identifying the most relevant features?

A) Core

B) Reduct

C) Boundary region

D) Indiscernibility relation

3. In Rough Set Theory, what does the lower approximation of a set represent?

A) Elements that possibly belong to the set

B) Elements that definitely belong to the set

C) Elements that do not belong to the set

D) Elements that belong to the set with some probability

4. Which of the following statements best describes a reduct in Rough Set Theory?

A) A subset of attributes that contains all the original attributes

B) A minimal subset of attributes that maintains the classification ability of the entire attribute set

C) A redundant set of attributes

D) A set of rules derived from the decision table

5. How does Rough Set Theory differ from Fuzzy Set Theory?

A) Rough Set Theory uses probabilistic methods, while Fuzzy Set Theory does not

B) Rough Set Theory deals with ambiguity through approximations, while Fuzzy Set Theory uses degrees of membership

C) Rough Set Theory is used for continuous data, while Fuzzy Set Theory is for categorical data

D) Rough Set Theory applies only to numerical data, while Fuzzy Set Theory applies to any type of data

5.8 Summing Up

This chapter delves into the intricacies of Rough Set Theory and Fuzzy Set Theory, focusing on how these mathematical frameworks handle uncertainty and vagueness in data analysis. The chapter begins with an introduction to Rough Set Theory, which was developed by Zdzisław Pawlak in the early 1980s. It explains the core concepts of lower and upper approximations, which are used to create a boundary region around uncertain data points. The lower approximation consists of elements that definitely belong to the set, while the upper approximation includes elements that possibly belong to the set. This method allows for the classification and analysis of imprecise data without the need for additional external information.

A significant part of the chapter is dedicated to the concept of reducts. A reduct is a minimal subset of attributes that can represent the essential characteristics of the entire dataset. The process of identifying reducts is crucial for feature selection, as it helps in reducing the dimensionality of the data while preserving its classification power. This reduction simplifies data analysis and improves the efficiency of subsequent computations.

The chapter also explores rule extraction methods in Rough Set Theory. The discernibility matrix is highlighted as a powerful tool for generating decision rules from data. These rules are essential for making informed decisions based on the patterns and relationships identified within the dataset.

In contrast, Fuzzy Set Theory is introduced as a means to handle uncertainty through the concept of partial membership. Unlike classical set theory, where elements either belong or do not belong to a set, Fuzzy Set Theory allows elements to have varying degrees of membership. This is represented by a membership function that assigns values between 0 and 1 to each element. This approach is particularly useful in situations where data is inherently ambiguous or where boundaries between classes are not well-defined.

The chapter further discusses the applications of Rough Set Theory and Fuzzy Set Theory in real-world scenarios, including their roles in feature selection, data mining, and decision-making processes. It highlights the advantages and limitations of each theory, providing insights into their practical implementation.

By comparing and contrasting Rough Set Theory and Fuzzy Set Theory, the chapter provides a comprehensive understanding of how these frameworks contribute to the analysis and interpretation of uncertain and imprecise data. It underscores the importance of these theories in the development of robust and efficient data analysis techniques, offering valuable tools for researchers and practitioners in various fields.

5.9 Answers to Check Your Progress

1. a 2.b 3.c 4.b 5.b

5.10 Possible Questions

1. What is the primary goal of Rough Set Theory?
2. Define the term "reduct" in the context of Rough Set Theory.
3. What is the purpose of the lower approximation in Rough Set Theory?
4. How does the upper approximation differ from the lower approximation in Rough Set Theory?
5. Explain the concept of an indiscernibility relation in Rough Set Theory.
6. What method is commonly used for rule extraction in Rough Set Theory?
7. How does Fuzzy Set Theory handle uncertainty compared to Rough Set Theory?
8. What is a membership function in Fuzzy Set Theory?
9. In what way can Rough Set Theory be used for feature selection?
10. Describe the main difference between a fuzzy set and a classical (crisp) set.
11. Explain the fundamental principles of Rough Set Theory and how it addresses uncertainty and vagueness in data. How do the concepts of lower and upper approximations contribute to this process?
12. Describe the process of identifying a reduct in Rough Set Theory. What is the significance of a reduct, and how does it

help in simplifying the data while preserving essential information?

13. Discuss the role of the indiscernibility relation in Rough Set Theory. How does this relation lead to the formation of equivalence classes, and what impact does it have on the approximation of sets?

14. Compare and contrast the methods of rule extraction in Rough Set Theory and other data mining techniques. What advantages does the discernibility matrix method offer in generating decision rules?

15. How does Fuzzy Set Theory extend classical set theory to handle uncertainty and imprecision? Provide examples to illustrate how membership functions assign degrees of membership to elements in fuzzy sets.

16. In Rough Set Theory, what are the key steps involved in constructing decision rules from a given decision table? How are these rules validated and applied to new data instances?

17. Explore the application of Rough Set Theory in feature selection. How does the identification of core and reduct attributes aid in dimensionality reduction, and what benefits does this bring to data analysis?

18. Discuss the similarities and differences between Rough Set Theory and Fuzzy Set Theory in their approach to handling uncertain and imprecise information. How do their respective methods of approximation and membership function contribute to decision-making processes?

19. Explain the significance of the boundary region in Rough Set Theory. How does the boundary region relate to the concepts of lower and upper approximations, and what information does it provide about the data?

20. Analyze the challenges and limitations of applying Rough Set Theory and Fuzzy Set Theory in real-world data analysis scenarios. What are some of the common issues faced, and how can these theories be adapted to address them effectively?

---×---

UNIT: 6
WEB MINING

Unit Structure:

6.1 Introduction

In this unit we are going to learn about Web mining. Web mining serves as a powerful tool for extracting, analyzing, and interpreting valuable information from the vast repository of web data. Its applications across diverse industries underscore its importance in shaping business strategies, improving user experiences, and gaining competitive advantage in the digital age.

6.2 Objectives

After going through this unit, you will be able to

- Understand web mining, web content mining, web usage mining, web structure mining and text mining
- Understand the key aspects of all the above mining techniques
- Understand the applications of all the above mining techniques
- Understand the benefits and impact of each of the mining types

6.3 Web Mining

Web mining refers to the process of using data mining techniques to extract valuable insights and information from web data, including web documents, web content, hyperlinks, and usage logs. It encompasses a broad range of activities aimed at discovering patterns and trends in the vast amount of information available on the web. Its applications span various industries, including search engines, e-commerce, market research, personalization, web analytics, fraud detection, and social network analysis. By leveraging web mining techniques, organizations can gain valuable insights, enhance user experience, and make informed business decisions. This approach encompasses a diverse set of activities aimed at uncovering patterns, trends, and relationships within web documents, content, hyperlinks, and usage logs.

6.3.1 Key Aspects of Web Mining

The key aspects of Web mining are:

1. **Data Sources and Types:** Web mining draws data from various sources such as web pages, social media platforms, online forums, and e-commerce sites. It encompasses both structured (e.g., databases) and unstructured (e.g., text, images) data types, requiring specialized techniques for extraction and analysis.

2. **Techniques and Methods:** Several data mining techniques are applied in web mining:
    - **Content Mining:** Analyzing the textual and multimedia content of web pages to extract relevant information. This involves techniques like text mining for sentiment analysis, topic modeling, and information extraction.
    - **Structure Mining:** Examining the linkage structure of the web, including hyperlinks between pages. Methods like link analysis, Page Rank algorithms, and network analysis help understand web connectivity and authority ranking.
    - **Usage Mining:** Studying user behavior and interaction patterns on websites to optimize user experience, personalize content, and improve site navigation.

6.3.2 Applications

The applications of web mining are:

- **Search Engines:** Enhancing search relevance and ranking algorithms to deliver more accurate and useful search results.
- **E-commerce:** Analyzing customer behavior, preferences, and purchase patterns to personalize recommendations and improve product offerings.
- **Market Research:** Gathering insights into consumer sentiment, trends, and competitor analysis from social media and online forums.
- **Web Analytics:** Monitoring website traffic, user engagement metrics, and conversion rates to optimize marketing strategies and site performance.
- **Fraud Detection:** Identifying suspicious activities and patterns in financial transactions or online activities to prevent fraud and ensure security.
- **Social Network Analysis:** Studying relationships and influence among users in social networks to predict behavior and trends.

6.3.3 Benefits and Impact

By leveraging web mining techniques, organizations can derive actionable insights that drive informed decision-making and strategy development. These insights not only improve operational efficiency but also enhance customer satisfaction through personalized services and targeted marketing campaigns.
In summary, web mining serves as a powerful tool for extracting, analyzing, and interpreting valuable information from the vast repository of web data. Its applications across diverse industries underscore its importance in shaping business strategies, improving user experiences, and gaining competitive advantage in the digital age.

6.4 Web Content Mining

Web content mining is the process of extracting useful information from the content of web pages. This includes not only text but also images, audio, video, and other multimedia data. The primary goal of web content mining is to discover patterns, trends, and meaningful relationships within the unstructured data that exists on the web, transforming it into structured information that can be more easily utilized for various applications. This field employs a variety of techniques and tools from natural language processing (NLP), machine learning, and information retrieval to analyze and interpret web content. For instance, NLP can be used to understand and derive meaning from text data, identifying keywords, topics, sentiments, and entities. Machine learning algorithms can classify and cluster web pages based on their content, helping in the organization and retrieval of information. Applications of web content mining are vast and diverse. It can be used to improve search engine results by providing more relevant responses to user queries. It helps in generating recommendations on e-commerce sites by analyzing product descriptions and user reviews. Web content mining is also valuable for market research, as it can provide insights into consumer behavior, preferences, and emerging trends by analyzing online discussions, social media posts, and other web-based communications.

Web content mining transforms the vast, diverse, and unstructured data available on the web into structured and useful information, enhancing our ability to understand and utilize web content effectively for various purposes, from improving user experiences to supporting business decision-making.

6.4.1 Key Aspects of Web Content Mining

The key aspects of Web Content Mining are:

1. **Data Types and Sources:** Web content mining deals with a wide array of data types, including textual content, images, videos, audio files, and other multimedia elements. These data sources are gathered from web pages, social media platforms, online forums, digital libraries, and other online repositories.
2. **Techniques and Methods:**
    - **Natural Language Processing (NLP):** NLP techniques are fundamental for analyzing and understanding textual content. They include tasks like tokenization, part-of-speech tagging, named entity recognition, sentiment analysis, topic modeling, and summarization. These techniques enable the extraction of keywords, entities (such as people or organizations mentioned), and sentiment from text.
    - **Machine Learning Algorithms:** Supervised and unsupervised machine learning algorithms play a crucial role in web content mining. They can be used for text classification (e.g., categorizing web pages into topics or genres), clustering similar documents, and building recommendation systems based on user behavior and content analysis.
    - **Image and Video Analysis:** Techniques such as image recognition, object detection, and video

summarization are employed to extract meaningful information from multimedia content on the web.

6.4.2 Applications

The main applications of web content mining are:

- **Search Engines Enhancement:** Web content mining improves the relevance and accuracy of search engine results by analyzing and indexing web pages based on their content. This ensures that users receive more precise responses to their queries.
- **E-commerce Recommendations:** By analyzing product descriptions, customer reviews, and user behavior, web content mining helps in generating personalized product recommendations. This enhances user satisfaction and increases sales conversion rates.
- **Market Research and Consumer Insights:** Web content mining provides valuable insights into consumer behavior, preferences, and market trends. It analyzes online discussions, social media posts, and reviews to identify emerging trends, sentiment towards products or brands, and potential customer needs.
- **Content Organization and Retrieval:** By categorizing and structuring web content, mining techniques facilitate efficient information retrieval. This is crucial for digital libraries, academic research, and any platform requiring organized access to vast amounts of online information.

6.4.3 Benefits and Impact

The transformation of unstructured web data into structured information through content mining enhances decision-making processes in various domains. It empowers businesses to tailor their strategies based on comprehensive insights derived from web content analysis, leading to improved operational efficiency and competitive advantage.

In essence, web content mining plays a pivotal role in making sense of the vast and heterogeneous data available on the web. By employing advanced techniques from NLP, machine learning, and information retrieval, it facilitates deeper understanding, effective utilization, and strategic leveraging of web content across diverse applications and industries.

Check Your Progress

1. _________ is the textual and multimedia content of web pages to extract relevant information
2. What is Usage Mining?
3.Mention two applications of web mining.
4. ____________ and unsupervised machine learning algorithms play a crucial role in web content mining
5.What is the benefit of web content mining?

6.5 Web Usage Mining

Web usage mining involves the process of extracting useful information from web server logs, which record user interactions with a website. This type of mining focuses on analyzing patterns and behaviors of users as they navigate through a website. The goal is to understand how users interact with a website, which pages they visit, how much time they spend on each page, the sequence of pages they follow, and their overall browsing patterns. Web usage mining can be used to improve website design, enhance user experience, personalize content, and optimize website performance. Techniques such as clustering, association rule mining, and sequential pattern mining are often employed to analyze web usage data. The insights gained from web usage mining can help in identifying popular pages, detecting navigational bottlenecks, understanding user preferences, and improving the effectiveness of web-based marketing strategies. By leveraging web usage mining, organizations can make data-driven decisions to enhance the usability and functionality of their websites, ultimately leading to increased user satisfaction and engagement.

Web usage mining is a powerful technique that focuses on analyzing user interactions and behaviors on websites, using data typically gathered from web server logs.

6.5.1 Key Aspects of Web Usage Mining

The key aspects of web usage mining are:

1. **Data Sources and Collection:** Web usage mining relies on data collected from web server logs, which record various aspects of user interactions such as page views, clicks, session durations, entry and exit pages, IP addresses, and user agents. This data provides a detailed trail of how users navigate and interact with a website.
2. **Types of Analysis Techniques:**
    - **Clustering:** Grouping users into segments based on similar browsing behaviors. This helps in understanding different user personas and tailoring website features or content to meet their specific needs.
    - **Association Rule Mining:** Discovering relationships between pages that are frequently accessed together by users. This can suggest patterns in navigation or content consumption that influence user behavior.
    - **Sequential Pattern Mining:** Identifying sequences of pages visited by users in a specific order. This helps in understanding typical user paths through the website and optimizing the navigation flow.
    - **Sessionization:** Segmenting user interactions into sessions to analyze patterns within individual browsing sessions. This can reveal insights into session duration, page transitions, and user engagement within a single visit.

6.5.2 Applications

The applications of web usage mining are:

- **Website Design and Optimization:** Web usage mining provides insights into user preferences and behaviors that can inform website redesigns, layout improvements, and navigation enhancements to improve user experience.
- **Personalization and Content Recommendation:** By analyzing user preferences and browsing patterns, web usage mining enables personalized content recommendations and targeted marketing campaigns. This enhances user engagement and conversion rates.
- **Performance Optimization:** Identifying performance bottlenecks such as slow-loading pages or high bounce rates helps in optimizing website performance and enhancing overall usability.
- **User Behavior Analysis:** Understanding user intent and preferences through web usage mining assists in tailoring content, products, and services to meet user expectations, ultimately driving customer satisfaction and loyalty.

6.5.3 Benefits and Impact

Leveraging insights from web usage mining allows organizations to make informed, data-driven decisions to optimize their websites for better usability, increased user engagement, and improved business outcomes. By understanding and responding to user behavior effectively, businesses can achieve competitive advantages and enhance their online presence.

In summary, web usage mining is instrumental in extracting actionable insights from web server logs to understand user behaviors, optimize website functionality, and enhance overall user satisfaction. Its applications span across various industries, offering significant benefits in improving website performance, personalizing user experiences, and driving business growth through informed decision-making.

Check Your Progress

6. What is association rule mining?
7. What is sessionization?
8. Mention two applications of web usage mining.

6.6 Web Structure Mining

Web structure mining is a process of discovering and extracting useful information and patterns from the structure of web pages and the links between them. It involves analyzing the topology of the web, including how web pages are interconnected through hyperlinks, to uncover valuable insights.

6.6.1 Types of web structure mining

There are generally three main types of web structure mining techniques:

1. **Link Analysis**: This technique focuses on analyzing the graph formed by web pages and their interlinking structure. It can involve:
    - **Page Rank Algorithm**: Used by search engines to rank web pages based on the number and quality of links pointing to them.
    - **HITS Algorithm (Hyperlink-Induced Topic Search)**: Identifies authorities (pages with lots of incoming links) and hubs (pages that link out to authorities).
2. **Web Structure Mining for Information Retrieval**: This approach uses the structure of the web to improve information retrieval tasks. It includes techniques like:
    - **Clustering and Classification**: Grouping web pages based on their structural similarities.
    - **Ontology-based Mining**: Using ontologies to categorize and organize web content hierarchically.
3. **Web Usage Mining**: Although not directly structural, it often involves analyzing patterns of user behavior on a website, which can be influenced by the structure of the site (e.g., how users navigate through linked pages).

Overall, web structure mining aims to improve web navigation, search engine effectiveness, and user experience by leveraging the inherent structure and relationships within the web.

6.7 Text Mining

Text mining, particularly when based on web mining, involves extracting valuable information and insights from textual content found on the web. Here’s how it typically works within the context of web mining

1. **Data Collection**: Text mining starts with the collection of textual data from web pages. This can include articles, blogs, product reviews, social media posts, and more. Web crawlers and scrapers are commonly used to gather this large volume of data from diverse sources.
2. **Preprocessing**: Once data is collected, preprocessing steps are applied to clean and prepare the text for analysis. This includes tasks such as:
    - **Tokenization**: Breaking text into individual words or tokens.
    - **Normalization**: Converting text to lowercase, removing punctuation, and handling special characters.
    - **Stop word Removal**: Filtering out common words (e.g., "and", "the") that don’t carry significant meaning.
    - **Stemming or Lemmatization**: Reducing words to their root forms to consolidate variations (e.g., "running" to "run").
3. **Text Mining Techniques**: Several techniques are applied to analyze the processed text data:
    - **Information Extraction**: Identifying and extracting structured information such as named entities (e.g., people, organizations) and relationships from unstructured text.
    - **Sentiment Analysis**: Determining the sentiment expressed in text (positive, negative, neutral) to gauge public opinion or customer feedback.

- **Topic Modeling**: Discovering topics or themes within a collection of documents to understand the main subjects discussed.
- **Text Classification**: Categorizing text into predefined classes or categories based on its content (e.g., spam detection, genre classification).
- **Keyword Extraction**: Identifying key phrases or terms that are most relevant to a particular document or set of documents.

4. **Applications**: Text mining based on web data finds applications in various fields:
    - **Business Intelligence**: Analyzing customer feedback, market trends, and competitor analysis.
    - **Healthcare**: Analyzing medical literature and patient records for insights into treatments and diseases.
    - **Social Media Analysis**: Understanding public opinion and trends based on posts and comments.
    - **E-commerce**: Product reviews and customer sentiment analysis for improving products and services.

In summary, text mining within the realm of web mining leverages techniques to extract, analyze, and interpret textual data from the web to derive valuable insights and support decision-making in various domains.

6.8 Summing Up

1. Web mining aims to extract valuable insights and information from web data, including web documents, content, hyperlinks, usage logs, and user behavior.
2. It helps organizations understand web dynamics, enhance user experience, optimize websites, personalize content, improve marketing strategies, and make informed business decisions.
3. Web content mining focuses on extracting meaningful information from web pages, including text, images, videos, and multimedia data.

4. It uses natural language processing (NLP), machine learning, and information retrieval techniques to analyze and interpret content. Tasks include sentiment analysis, topic modeling, keyword extraction, and information extraction.
5. Web Content mining helps in search engine results, generates product recommendations, supports market research, and improves content organization.
6. Web usage mining analyzes user interactions and behaviors on websites using web server logs. It includes clustering, association rule mining, sequential pattern mining, and sessionization to understand user navigation patterns, preferences, and engagement.
7. It optimizes website design, improves user experience, personalizes content recommendations, and enhances website performance by identifying bottlenecks and improving usability.
8. Web structure mining examines the link structure of the web, focusing on how web pages are interconnected through hyperlinks.
9. It involves link analysis, PageRank algorithms, and network analysis to determine page importance, authority, and connectivity patterns.
10. Web structure mining also enhances search engine algorithms, improves navigation systems, and supports knowledge discovery by uncovering relationships and hierarchies within web data.
11. Text mining extracts useful information and patterns from textual data, including web content, documents, and social media posts.
12. Text mining utilizes NLP techniques such as tokenization, part-of-speech tagging, entity recognition, sentiment analysis, and topic modeling.

13. It supports information retrieval, sentiment analysis, document summarization, content categorization, and knowledge discovery from unstructured text sources.
14. Each of these topics play a crucial role in leveraging web data and textual content to derive insights that inform decision-making, enhance user experiences, and optimize digital strategies across various domains and industries.

6.9 Answers to Check Your Progress

1. Content Mining

2. Usage Mining is the study of user behavior and interaction patterns on websites to optimize user experience, personalize content, and improve site navigation.

3. The two applications of web mining are:

Market Research: Gathering insights into consumer sentiment, trends, and competitor analysis from social media and online forums.

Web Analytics: Monitoring website traffic, user engagement metrics, and conversion rates to optimize marketing strategies and site performance

4. Supervised

5. The benefit of web content mining is that it plays a pivotal role in making sense of the vast and heterogeneous data available on the web. By employing advanced techniques from NLP, machine learning, and information retrieval, it facilitates deeper understanding, effective utilization, and strategic leveraging of web content across diverse applications and industries.

6. Association Rule Mining is discovering relationships between pages that are frequently accessed together by users. This can

suggest patterns in navigation or content consumption that influence user behavior.

7. Sessionization **is** segmenting user interactions into sessions to analyze patterns within individual browsing sessions

8. The two applications of web usage mining are:

Website Design and Optimization: Web usage mining provides insights into user preferences and behaviors that can inform website redesigns, layout improvements, and navigation enhancements to improve user experience.

Personalization and Content Recommendation: By analyzing user preferences and browsing patterns, web usage mining enables personalized content recommendations and targeted marketing campaigns. This enhances user engagement and conversion rates.

6.10 Possible Questions

1. What are the main forms of content that can be analyzed in web content mining?
2. Explain how natural language processing is utilized in web content mining.
3. Describe some practical applications of web content mining in business and technology.
4. What is the primary source of data used in web usage mining?
5. How can web usage mining help improve user experience on a website?
6. What are some common techniques used in web usage mining?
7. What is web structure mining, and how does it relate to analyzing web pages?
8. Name two key techniques used in web structure mining. Briefly explain how each technique contributes to understanding web content.

9. How does the PageRank algorithm work, and what is its significance in web structure mining?
10. Explain the difference between link analysis and web usage mining. How are they related to understanding user behavior on the web?
11. What is text mining, and how does it differ from traditional data mining techniques?
12. List three preprocessing steps commonly used in text mining. Explain why each step is important.
13. Describe two major techniques used in text mining. Provide an example of how each technique can be applied to analyze web-based textual data.
14. Discuss two real-world applications of text mining based on web data. How does text mining contribute to insights in these applications?
15. What is the objective of web mining?
16. How does web structure mining differ from web content mining?
17. Explain the role of natural language processing (NLP) in web content mining.
18. How can web content mining enhance e-commerce platforms?
19. What are the main techniques used in web usage mining?
20. How does web usage mining contribute to improving website performance?
21. What are the key tasks performed in text mining?
22. How can text mining benefit market research?

6.11 References and Suggested Readings

1. Pujari, A. K. (2001). *Data Mining Techniques*. Universities press.

---×---

UNIT: 7
TEMPORAL DATA MINING

Unit Structure:

7.1 Introduction

Temporal Data Mining:

The process of extraction of non-trivial, implicit, and potentially essential data from large sets of temporal data is known as the Temporal data mining. Temporal data mining are generally numerical values and a series of primary data types, and it deals with collecting beneficial knowledge from temporal data.

The interest in finding hidden data has shattered in the last decade with the extend of stored data. Classifying data, finding relationships, and data clustering are used to finding hidden data. Treating data with temporal dependencies is the major drawback that comes during the discovery process. Temporal data can be considered as an unordered collection of data by most of the data mining techniques and ignoring temporal data.

7.2 Objectives

This unit is an attempt to analyse the ideas of Temporal data mining. After going through this unit, you will
be able to-
• explain Temporal data mining.
• discuss objectives of Temporal data mining.
• explain concept of various tasks in temporal mining.

To find the temporal patterns, unexpected trends, or several hidden relations in higher sequential data are the aim of temporal data mining, which are composed of a sequence of nominal symbols from the alphabet known as temporal sequence and a sequence of continuous real-valued components called a time series, by using a set of approaches from database technologies, statistics, and machine learning.

Description of temporal data, representation of similarity measures, and mining services are used to form Temporal data mining.

Temporal Data Mining includes processing time series, generally orders of data, which calculate values of the similar attribute at an order of many time points. Pattern matching using such data, where specific patterns of interest are searching, has captivated considerable interest in current years.

Exploitation of efficient techniques of data storage, quick processing, and quick retrieval methods are used by temporal data mining that have been advanced for temporal databases.

Temporal data is a temporal data mining algorithm. The process of knowledge discovery in temporal databases temporal data mining is an individual phase that calculate temporal patterns from or fit models too, Temporal data mining is concerned with the analysis of temporal data and for discovering temporal patterns and consistencies in sets of temporal information. Temporal data mining also allows the possibility of computer-driven, automatic exploration of the data. There are various tasks in temporal data mining which are as follows −

- Data characterization and comparison

- Clustering analysis
- Classification
- Association rules
- Pattern analysis
- Prediction and trend analysis

Temporal data mining, a new way of interacting with a temporal database and specifying queries at a much more abstract level temporal structured query language permit. Due to multiple and multi-dimensionality temporal data mining facilities data exploration for problems.

The basic goal of temporal classification is to predict temporally related fields in a temporal database based on other fields. The problem as deciding the general value of the temporal variable being predicted given the different fields, the training data in which the target variable is given for each observation, and a set of assumptions representing one's prior knowledge of the problem. Complex problem of density estimation is associated with the temporal classification techniques.

The temporal relationship between items are mine by most temporal association rule mining algorithms. Though, the sequential relations between successive items are not known.

STOP TO CONSIDER

Temporal refers to time.

7.3 Temporal association rules

A temporal association rule states that a set of items tend to appear with another set of items in the same transactions, within a specific time slots.

By searching data for frequent if-then patterns association rules are created. To identify the most important relationships, support and confidence are used. Support is sign of how often the items appear in the data.

Different types of association rules in data mining include:

- Multi-relational association rules

- Generalized Association rule
- Interval Information Association Rules
- Quantitative Association Rules

A temporal association rule that holds during specific time slots. Temporal association rule explain show a set of items try to appear along with another set of items in the same transactions, in a particular time.

For example, the rule "bread and butter are frequently sold together in morning hours" is a temporal association rule. Another example is the rule "mask → gloves", which could be extracted from the equipment repository for a hospital. This rule indicates that most medical staff who wear a mask also wear gloves.

In a real application to mine more significant temporal association rules, we need to know:

- The frequency that different time series with same time interval occur
- The frequency that some status of one time series occurs behind the certain status of another time series

Most temporal association rule mining algorithms can mine the temporal relationship between items, but the sequential relations between successive items remain are not known, while the discovering sequential relationship between successive items are done by sequential pattern mining algorithms.

Here are some main concepts related to temporal association rules:

Temporal database

Temporal databases store information about the changes in a system or time of occurrence of events in where temporal association rules are often applied.

Time Interval

Temporal association rules consider time gap or intervals between events. There are two types of time interval one is explicit (e.g., timestamps) and another one is implicit (e.g., ordering of events).

Events and Sequences
Events are occurred with associated timestamps in a temporal context. To discover patterns and associations sequences of events over time are analysed.

Temporal association rule
It expresses relationships between events that occur within specific time gap. It is represented in the form of if for antecedent events that are occur within time interval, then consequent events which are occur within time interval.

Support and confidence
In the dataset frequency of a temporal pattern is measured by Support and the probability that the consequent events will occur given the antecedent events is measured by confidence. These metrics are adapted to the temporal context.

Sequential patterns
Temporal association rules find the sequential patterns, where events are not only associated but also follow a specific order over time.

Applications
Temporal association rules are applied in various fields like finance, healthcare, and retail.

Algorithm
Different algorithms, often extensions of traditional association rule mining algorithms, have been developed to discover temporal association rules. These algorithms take into account the temporal order of events.

Challenges
Temporal data analysing introduces challenges like, dealing with data gaps, handling time intervals, and managing the figuring complexity of mining temporal patterns.

Tool support
For discovering temporal association rules various data mining tools and platforms uses support, allowing analysts to explore time-dependent patterns in their datasets.

7.4 Sequence Mining

The mining which is used to uncovers patterns within ordered datasets is called sequence mining and it uncover such as hidden correlations and predicting future trends. It's useful in fields such as genomics, web analytics, market analysis, transforming raw data into actionable insights etc.

Sequence mining is a structured data mining in where administrator and the database check sequences or trends in the data. Sequence mining is split into two fields. One is Item set sequence mining which is used in areas like marketing and another one is string sequence mining is used by biology research. Sequence mining is not similar as regular trend mining, because the sequence data are more specific, which is used to building an effective database and it is difficult for database designers, and it can sometimes go awry if the sequence is any different from the common sequence.

At any point, all types of databases are used to mine for data. This mining helps some research parties and businesses to find something they need. They are looking for some sort of trend, but what the trend that is and it depend on the database design for how the information is specific. Database is design to find very specific sequence in sequence mining. There will be no variation in sequence mining. This is a structured data mining of unique form in where the database looking for similarities of structured data.
Sequence data mining is of two categories. One is item set data mining which is used in marketing and business to find specific trends in product types, product placement, sales number in a store and the use of a product. These figures also applied in marketing algorithms to help strategize marketing project which is used to bolster sales. Information about a product and how it does typically is taken from the database, but the defining aspect of item set sequence mining is that the sequence is taken from multi-symbol database cells.

Sequence data in sequence data mining are useful in a variety of fields including Web Click-stream Analysis, marketing which is already proven. E.g. A sequence s is a set of ordered things indicated by <s1,s2,s2….sn>. In activity recognition problems, timestamps are also used along to arrange the series. To find interesting patterns in sequence mining in data based on some subjective or objective evaluation of how interesting it is.

Discovering all the frequent Sequence Data in Data Mining is a difficult. Due to the combinatorial and exponential search space, it can be difficult. Several sequence mining approaches have been published in the last decade that use various heuristics to handle the exponential search. GSP (Generalized Sequential Pattern) which was based on the a priori approach for mining frequent item sets was the first sequence mining algorithm for Sequence Data in Data Mining. GSP goes over the database numerous times to count the support for each sequence and generate candidates. The sequences with a support count less than the minimum support are pruned.

Sequence Data in Data Mining Types:

A Sequence is a list of events in chronological order. Based on the characteristics of the events Sequence Data in Data Mining can be classified into three kinds:

- **Sequence Data in Data Mining:** Time-Series Data Similarity Search
- **Sequence Data in Data Mining:** Time-Series Data Regression and Trend Analysis
- **Sequence Data in Data Mining:** Sequential Pattern Mining in Symbolic Sequences Data in Data Mining: Time-Series Data Similarity Search

It is a collection of integer values collected over a period under Sequence Data in Data Mining. The values are usually considered at regular time intervals (such as each minute, hour, or day).

Stock market analysis, economic and sales forecasting, budgetary analysis, utility studies, inventory studies, income predictions, workload projections, process, and quality service are just a few of the applications for time-series databases in Sequence Data in Data Mining. Natural events, mathematics, technical investigations, and pharmacological therapies all benefit from them.

Time-Series Data Regression and Trend Analysis:

In the application of Data and Signal Analysis in Sequence Data in Data Mining, Regression Analysis of time-series data has been designed highly. To define time-series data, trend analysis creates an integrated model using the four primary aspects or motions shown below in Sequence Data in Data Mining.

- Trend or long-term movements define the overall direction in which a time-series graph changes over time in Sequence Data in Data Mining, for example, finding trend curves including the dashed curve using the weighted moving average and the least-squares approach.
- Long-term vibrations around a trend line or curve are known as cyclic motions in Sequence Data in Data Mining.
- Seasonal variations are trends that a time series follows during similar seasons of multiple years, such as holiday shopping seasons. The data has to be "seasonalized" using an autocorrelation-based seasonal index for effective trend analysis in Sequence Data in Data Mining.
- Random movements are sporadic changes in an organization caused by chance occurrences such as labour conflicts or stated personnel changes.

Sequence Data in Data Mining:

An organized series of items or occurrences, documented with or without a specific idea of time, makes up a symbolic sequence in Sequence Data in Data Mining. In research and engineering, as well as natural and social developments, data of symbolic series can be used in a variety of ways, including user purchasing sequences, online click streams, software implementation sequences, biological sequences, and event sequences.

Because biological sequences have a complex semantic meaning and pose several difficult research issues, most researches focus on bioinformatics.

Biological Sequence Alignment:

Nucleotide or amino acid sequences are biological sequences. Biological sequence analysis compares, aligns, indexes, and studies biological sequences, making it important in bioinformatics and contemporary biology.

The fact that all living entities are linked through development makes sequence alignment possible. Nucleotide (DNA, RNA) and protein sequences of species that are closer in evolution must have more in common. An alignment is a process of aligning sequences to achieve a maximum level of identity, which also determines the degree of resemblance between them.

Stop to Consider

- Time series data mining makes our natural ability to visualize the shape of real-time data.
- Web access pattern analysis, weather prediction, production processes, and web intrusion detection.

7.5 Time Series

Time Series is a data mining technique that forecasts target value based solely on a known history of target values. It is a specialized form of Regression, known in the literature as auto-regressive modelling. The input to time series analysis is a sequence of target values.

Time Series Data:

Time series data is a collection of observations or measurements taken at successive, equally spaced intervals over time. Examples include stock prices, temperature readings, sensor data, and more.

Patterns in Time Series:

To identify patterns within the data, such as trends, seasonality, anomalies, and other recurring structures is the aim of time series. For providing valuable insights for prediction and decision-making one must understand these patterns.

Temporal patterns:

Recurring behaviours or structures in the time series data are known as Temporal patterns. Patterns may be short-term or long-term and may include trends, cycles, or irregularities.

Time series analysis technique:
The techniques are used for time series analysis, including statistical methods, machine learning algorithms, and signal processing techniques. Common approaches include autoregressive integrated moving average (ARIMA) models, exponential smoothing methods, and machine learning algorithms like recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks.

Feature extraction
Extracting relevant features from time series data is crucial for effective analysis. Features may include statistical measures, frequency domain features, or other characteristics that capture important aspects of the temporal behaviour.

Anomaly detection
Anomaly detection in time series data involves identifying cases that differ very much from the expected patterns. This is crucial for detecting uncommon events or outliers.

Forecasting and prediction
For forecasting future values based on historical data time series mining is used. Predictive models can be trained to anticipate future trends or events.

Seasonality and trends
Seasonality means periodic patterns that repeat at regular intervals of time, while trends give the long-term changes in the data. To identifying and understanding this type of components is important for making informed predictions.

Applications
The applications of time series mining are finance (stock market predictions), healthcare (patient monitoring), environmental monitoring, and manufacturing (predictive maintenance).

Tools and Libraries:
Several tools and libraries, such as Python's pandas, scikit-learn, and Tensor Flow, provide functionalities for time series analysis and mining. Specialized tools like R and specialized libraries like Stats models are also commonly used.

Challenges:
Challenges in time series mining include dealing with missing data, handling irregular sampling intervals, and selecting appropriate models for the specific characteristics of the time series.

Time series mining is a critical component in understanding and leveraging temporal patterns within data, providing valuable insights for decision-making and prediction in a wide range of applications.

Time series represents a collection of values or data obtained from the logical order of measurement over time. Time series data mining makes our natural ability to visualize the shape of real-time data. It is an ordered sequence of data points at uniform time intervals.

Time Series Analysis comprises methods for analysing time-series data to extract meaningful statistics, rules, and patterns. These rules and patterns might be used to build forecasting models that are able to predict future developments.

Application of Time Series Mining:
1. Financial:
1.1 Used for stock price evaluation
1.2 For the measurement of Inflation

2. Industry:
2.1 Determine the power consumption

3. Scientific:
3.1 Used for experiment results

4. Meteorological:
4.1 Concerned with the processes and phenomena of the atmosphere, basically for forecasting weather

Characteristic of time series components:
1. Trend
2. Cycle

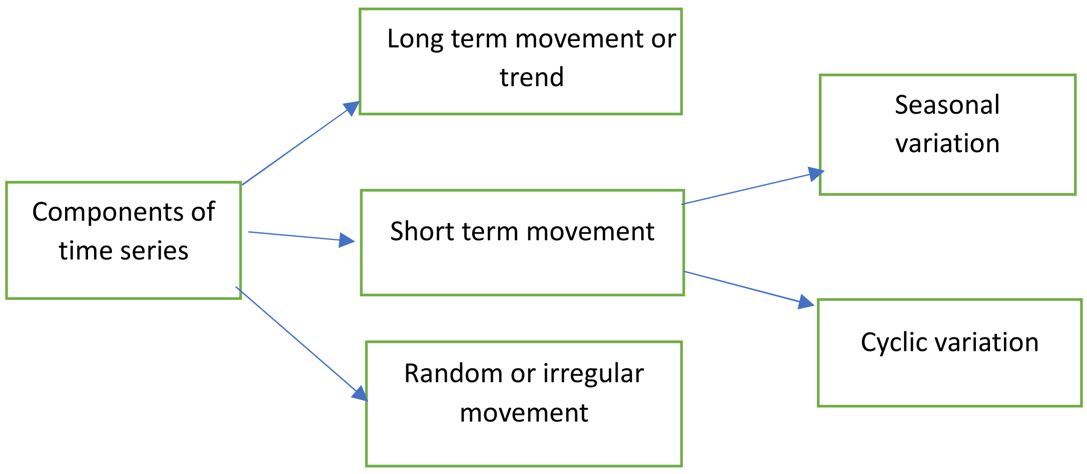
3.Seasonal
4. Irregular

Category of Time-Series Movements:
1. Long-term or trend movements:
The general direction in which a time series is moving over a long interval of time. It shows the general tendency of the data to increase or decrease a long period of time.

2. Cyclic movements or cycle variations:
Long term oscillations about a trend line or curve. For example, business cycles. This oscillatory movement has a period of oscillation of more than a year.

3. Seasonal movements or seasonal variations:
Almost identical patterns that a time series appears to follow during corresponding months of successive years. This variation will be present in a time series if the data are recorded hourly, daily, weekly, or monthly.

4. Irregular or random movements:
These fluctuations are unforeseen, uncontrollable, and unpredictable. They are not regular variations and are purely random or irregular.


Long term movement or trend
Seasonal variation
Components of time series
Short term movement
Cyclic variation
Random or irregular movement

Components for Time Series Analysis

7.6 Pattern Detection

Pattern mining identifies rules that describe patterns within data. The first, termed "pattern detection," consists of examining an arbitrary set of figures and selecting those having some specified form. The second problem, "pattern recognition," consists of identifying a given figure which is known to belong to one of a finite set of classes.

The pattern recognition process typically involves four phases:

1. Preprocessing: Preparing the data for analysis
2. Training
3. Testing
4. Deployment

Neural pattern recognition is a method that uses artificial neural networks (ANN) to detect patterns. ANNs are computational systems that can learn to recognize patterns in various data types, such as textual, visual, or audio.

There are three main types of pattern recognition:

- Statistical
- Structural (or syntactic)
- Neural

Pattern recognition has a variety of applications, including image processing, speech and fingerprint recognition, aerial photo interpretation, optical character recognition in scanned documents such as contracts and photographs, and even medical imaging and diagnosis.

Pattern mining concentrates on identifying rules that describe specific patterns within the data. Market-basket analysis, which identifies items that typically occur together in purchase transactions, was one of the first applications of data mining.

Stop to Consider

Time Series Analysis comprises methods for **analyzing time-series data in order to extract meaningful statistics, rules and** patterns.

Check Your Progress:

1) What do you mean by temporal data mining?
2) What are the objectives of temporal data mining?
3) What are the tasks of temporal data mining?
4) What is the temporal association rule?
5) What is an example of a temporal association?
6) What are temporal rules?
7) What is association rule used for?
8) What is an example of sequence data in data mining?
9) Which of the following sequence can be considered as the correct process of data mining?
10) What is Biological Sequence Alignment?
11) What is time series data mining technique?
12) What are the four 4 main components of a time series?
13) What are time series used for?
14) Is the database play a vital role in Time Series mining?

15) What are the types of pattern mining in data mining?
16) What are the four 4 main data mining techniques?
17) What methods does data mining use to find patterns and relationships in data?
18) What are the applications of pattern mining?
19) What are the different types of pattern?

7.8 Summing Up

- Temporal data mining is the process extraction of non-trivial, implicit, and potentially important data from huge sets of temporal data.
- It primarily focuses on classifying data, finding relationships, and data clustering.
- The major drawback during the discovery process is temporal dependencies.
- Temporal data mining is an individual phase.
- The aim of temporal data mining is to find temporal patterns, unexpected trends and many more.
- The tasks of temporal mining which are as follows −

    - Data characterization and comparison
    - Clustering analysis

    - Classification
    - Association rules
    - Pattern analysis
    - Prediction and trend analysis
- Temporal association rules extend traditional association rules by using the temporal order of events.
- Data must be time-stamped, indicating when each transaction or event occurs.
- Events or transactions are occurring as timestamps.
- Temporal association rules often involve analysing sequences of events over time.
- Time intervals between events are the time to capture time duration between occurrences.
- Support measures the frequency of a temporal pattern, while confidence measures the reliability of the rule.
- Stock Market Analysis: Identify patterns in stock price movements.
- Healthcare Monitoring: Analyses patient records for temporal patterns in health events.
- Web Usage Mining: Understand user behaviour patterns on websites over time.
- Handling Irregular Time Intervals: Dealing with unevenly spaced time-stamped data.
- Data Preprocessing: Cleaning and aligning temporal data for analysis.
- Computational Complexity: Analysing large datasets with time-dependent rules.
- Temporal association rules can be visualized using timelines or sequence diagrams to understand the patterns.
- Incorporating machine learning and deep learning techniques for more accurate temporal pattern recognition.
- Sequence data mining focuses on discovering meaningful patterns and relationships within ordered sequences of data, considering the inherent order of events.

- Identifies frequent sequences or patterns of events within a dataset.
- Applies to sequences with symbolic representations, like DNA sequences, text documents, or financial transactions.
- Discovers rules that describe relationships between events in a sequence.
- Incorporates time-related information, such as the time gap between events or the duration of events.

- Time series data is a sequence of observations or data points collected and recorded over time. It can be used to analyse trends, patterns, and behaviours.
- Time series data is inherently ordered, with each data point associated with a specific time or time interval.
- Components of Time Series:
  - Trend: Long-term movement or direction in the data.
  - Seasonality: Regular and predictable fluctuations or patterns.
  - Cycle: Repeating up-and-down movements, not necessarily of fixed duration.
  - Irregularity/Noise: Unpredictable random variations.
- Applications:
  - Financial Forecasting: Predicting stock prices and market trends.
  - Economic Analysis: Monitoring economic indicators over time.
  - Weather Forecasting: Analysing temperature, precipitation, etc., for prediction.
- Time Series Analysis Techniques:
  - Descriptive Analysis: Summarizing key statistical measures and visualizing trends.
  - Smoothing Techniques: Removing noise to reveal underlying patterns.
  - Seasonal Decomposition: Breaking down time series into trend, seasonality, and residual components.
- Pattern detection in data mining involves identifying meaningful, often hidden, structures, trends, or relationships within large datasets.

- The primary goal is to discover valuable insights, correlations, and patterns that can inform decision-making or predictions.
- dataset and identify potential patterns.
- Types of Patterns:
  - Association Patterns: Discover relationships between variables in the form of rules.
  - Sequential Patterns: Identify patterns based on the sequential order of events.
  - Clustering Patterns: Grouping similar data points based on certain features.
- Application Areas: Pattern detection is applied across various domains, including marketing, finance, healthcare, and fraud detection.

7.9 Answer to check your progress/Possible Answers to SAQ

1) What do you mean by temporal data mining?

   Answer: The process of extraction of non-trivial, implicit, and potentially essential data from large sets of temporal data is known as the Temporal data mining.
2) What are the objectives of temporal data mining?

   Answer: The objective of temporal data mining is to find patterns, trends, and relations within data and to extract meaningful information from the data to show how the data trend has changed over time.
3) What are the tasks of temporal data mining?

   Answer: The tasks in temporal data mining are as follows −
   - Data characterization and comparison
   - Clustering analysis
   - Classification
   - Association rules
   - Pattern analysis
   - Prediction and trend analysis
4) What is the temporal association rule?

   Answer: A temporal association rule expresses that a set of items tends to appear along with another set of items in the same transactions, in a specific time.

5) Is the database play a vital role in Time Series mining?

Answer: The database is the collection of data retrieved from a different source in which the data are stored in a structural, non-structural format on their respective columns. Time Series database consists of a sequence of values or events changing with time. Data are recorded at regular intervals.

6) What is an example of a temporal association?

Answer: A temporal association rule is an association rule that holds during specific time intervals. An example is that bread and butter are frequently sold together in a shop in morning hours.

7) What are temporal rules?

Answer: A temporal decision rule (or simply a temporal rule) is a decision rule that can be used to predict or retrodict the value of a decision attribute, using condition attributes that are observed at times other than the decision attribute's time of observation.

8) What is association rule used for?

Answer: Association rule mining is a technique used to uncover hidden relationships between variables in large datasets. It is a popular method in data mining and machine learning and has a wide range of applications in various fields, such as market basket analysis, customer segmentation, and fraud detection.

9) What is an example of sequence data in data mining?

Answer: Examples of sequence data include DNA, protein, customer purchase history, web surfing history, and more. Sequence Data Mining provides balanced coverage of the existing results on sequence data mining, as well as pattern types and associated pattern mining methods.

10) Which of the following sequence can be considered as the correct process of data mining?

Answer: The correct order of the processes involved in the data mining process is Infrastructure, exploration, analysis, interpretation, and exploitation.

11) What is Biological Sequence Alignment

Answer: Nucleotide or amino acid sequences are biological sequences. Biological sequence analysis compares, aligns, indexes, and studies biological sequences, making it important in bioinformatics and contemporary biology.

12) What is time series data mining technique?

Answer: The Time Series mining function provides algorithms that are based on different underlying model assumptions with several parameters. The learning algorithms try to find the best model and the best parameter values for the given data. If you do not specify a seasonal cycle, it is automatically determined.

13) What are the four 4 main components of a time series?

Answer: Components for Time Series Analysis

- Trend.
- Seasonal Variations.
- Cyclic Variations.
- Random or Irregular movements.

14) What are time series used for?

Answer: In particular, a time series allows one to see what factors influence certain variables from period to period.

15) What are the types of pattern mining in data mining?

Answer: Types of Data mining include:

- Clustering.
- Prediction.
- Classification.
- Genetic Algorithms.
- Regression.
- Association rule learning.
- Anomaly detection.
- Artificial Neural Network Classification.

16) What are the four 4 main data mining techniques?

Answer: Data mining typically uses four data mining techniques to create descriptive and predictive power: regression, association rule discovery, classification, and clustering.

17) What methods does data mining use to find patterns and relationships in data?

Answer: Popular data mining techniques include the following types:

Association rule mining. In data mining, association rules are if-then statements that identify relationships between data elements.

- Classification.
- Clustering.
- Regression.
- Sequence and path analysis.
- Neural networks.

18) What are the applications of pattern mining?

Answer: Frequent Pattern Mining applications are applied to many different domains such as market basket and risk analysis in commercial situation epidemiology, clinical medicine, fluid dynamics, astrophysics, crime prevention, counter-terrorism, sale campaign analysis.

19) What are the different types of patterns?

Answer: Different types of patterns are:

- Association Patterns: Discover relationships between variables in the form of rules.
- Sequential Patterns: Identify patterns based on the sequential order of events.
- Clustering Patterns: Grouping similar data points based on certain features.

7.10 Possible Questions

1. In which domains is temporal association rule mining particularly valuable?
2. Explain the concept of support and confidence in the context of temporal association rules.
3. What is the significance of sliding windows in temporal association rule mining?
4. How does the temporal order of events contribute to the discovery of patterns in data sequences?
5. What emerging trends are impacting the field of temporal association rule mining?
6. What is sequence mining in the context of data mining?

7. How does sequence mining differ from traditional association rule mining?
8. What role does feature selection play in pattern detection?
9. Explain the concept of overfitting in the context of pattern detection.
10. How can unsupervised learning be applied to pattern detection?
11. What are some common challenges in detecting patterns in high-dimensional data?
12. How does the choice of distance metric impact pattern detection algorithms?
13. What is the importance of interpretability in pattern detection models?
14. What is the curse of dimensionality, and how does it relate to pattern detection?
15. In what ways can deep learning techniques enhance pattern detection tasks?
16. What are some ethical considerations in the use of pattern detection algorithms?
17. Explain the difference between rule-based and statistical approaches to pattern detection.
18. Can you provide an example scenario where sequence mining would be useful?
19. What are the key components of a sequence in sequence mining?
20. Name a popular algorithm used for mining sequential patterns.
21. What is the significance of support and confidence in sequence mining?
22. Explain the concept of lag or gap constraints in the context of sequence mining.
23. What are the applications of sequence mining in real-world scenarios?
24. How does the concept of symbolic sequences relate to sequence mining?
25. What challenges are associated with mining sequences from large datasets?
26. How does visualization aid in understanding sequential patterns?
27. In which domains is sequence mining particularly valuable?

28. How do you handle irregular time intervals when performing sequence mining?
29. What emerging trends are impacting the field of sequence mining?

7.11 References and Suggested Readings

- Data Mining techniques by Arun K. Pujari.
- www.javatpoint.com

---×---