Sample Questions:

Module 1:

1. What is DWH? Explain DWH characteristics.

- A data warehouse is a large collection of business data used to help an organization make decisions.
- It is a system used for reporting and data analysis and is considered a core component of business intelligence.
- DWs are central repositories of integrated data from one or more disparate sources. They store current and historical data in one single place that are used for creating analytical reports for workers throughout the enterprise.
- According to William H. Inmon, a leading architect in the construction of data warehouse systems, “A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management’s decision making process”

There are four key features of data warehouses —subject-oriented, integrated, time-variant, and nonvolatile:

- **Subject-oriented:** Unlike the operational systems, the data in the data warehouse revolves around the subjects of the enterprise. Subject orientation is not database normalization. Subject orientation can be really useful for decision-making. Gathering the required objects is called subject-oriented.
- **Integrated:** The data found within the data warehouse is integrated. Since it comes from several operational systems, all inconsistencies must be removed. Consistencies include naming conventions, measurement of variables, encoding structures, physical attributes of data, and so forth.
- **Nonvolatile:** The data in the data warehouse is read-only, which means it cannot be updated, created, or deleted (unless there is a regulatory or statutory obligation to do so).
- **Time-variant:** While operational systems reflect current values as they support day-to-day operations, data warehouse data represents a long time horizon (up to 10 years) which means it stores mostly historical data. It is mainly meant for data mining and forecasting. (E.g. if a user is searching for a buying pattern of a specific customer, the user needs to look at data on his current and past purchases.)

2. What are the advantages and applications of DWH?

Advantages of Data Warehouse (DWH):

- Data warehouse allows users to access critical data from the number of sources in a single place. Therefore, it saves user’s time of retrieving data from multiple sources.

- Data warehouse provides consistent information on various cross-functional activities. It also supports ad-hoc reporting and query.
- Data Warehouse helps to integrate many sources of data to reduce stress on the production system.
- Data warehouse helps to reduce total turnaround time for analysis and reporting.
- Restructuring and Integration make it easier for the user to use for reporting and analysis.
- Data warehouse stores a large amount of historical data. This helps users to analyze different time periods and trends to make future predictions.

Applications of Data Warehousing

Sector	Usage
Airline	It is used for airline system management operations like crew assignment, analysis of route, frequent flyer program discount schemes for passenger, etc.
Banking	It is used in the banking sector to manage the resources available on the desk effectively.
Healthcare sector	Data warehouse used to strategize and predict outcomes, create patient's treatment reports, etc. Advanced machine learning, big data enable data warehouse systems to predict illness.
Insurance sector	Data warehouses are widely used to analyze data patterns, customer trends, and to track market movements quickly.
Retain chain	It helps you to track items, identify the buying pattern of the customer, promotions and also used for determining pricing policy.
Telecommunication	In this sector, data warehouse is used for product promotions, sales decisions and to make distribution decisions.

3. Why is the ER model not suitable for DWH?What are the steps in dimensional modeling?

A dimensional model in the data warehouse is designed to read, summarize, and analyze numeric information like values, balances, counts, weights, etc. in a data warehouse.

In contrast, relation models are optimized for addition, updating, and deletion of data in a real-time Online Transaction System.

Dimensional models are used in data warehouse systems and not a good fit for relational systems.

A dimensional model contains the same information as the ER model but packages the data in a symmetric format whose design goals are easy understandability, query performance, and resilience to change.

ER modeling aims to optimize performance for transaction processing. It is also hard to query ER models because of the complexity; Therefore ER models are not suitable for high-performance retrieval of data.

Data warehouses contain huge information. Data can't be fetched by normal technique so it requires special techniques.

The Entity-Relationship (ER) model is primarily designed for conceptual modeling of data within a specific application domain. While it's great for representing entities, their attributes, and the relationships between them in a relational database context, it may not be the best fit for a Data Warehouse (DWH) environment for several reasons:

Complexity of Data: Data Warehouses often deal with large volumes of data from various sources. The ER model may not adequately capture the complexity of these data relationships, hierarchies, and aggregations.

Performance: ER models may not be optimized for query performance in a Data Warehouse environment. Data Warehouses typically require denormalization, aggregation, and other optimizations to support efficient querying and reporting, which may not be easily represented in an ER model.

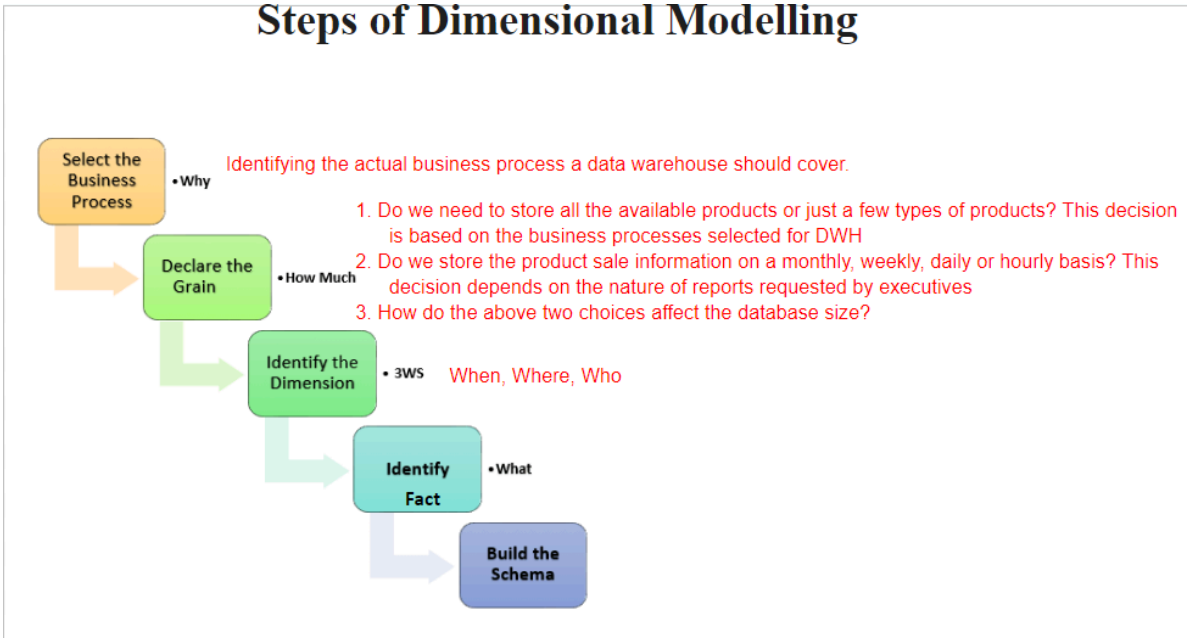
Historical Data: Data Warehouses often store historical data for analysis and reporting purposes. The ER model may not have built-in constructs to handle slowly changing dimensions, temporal data, or versioning, which are common requirements in Data Warehousing.

Multi-dimensional Modeling: Data Warehouses commonly use multi-dimensional modeling techniques such as star schemas or snowflake schemas to organize data for analytical purposes. These models are more specialized for data analysis and reporting than the generic entity-relationship model.

Data Integration: ER models are not inherently designed for integrating data from disparate sources, which is a key requirement in Data Warehousing. Data Warehouse environments often involve data integration, transformation, and cleansing processes, which may not be well-represented in an ER model.

Aggregation and Summarization: Data Warehouses often require pre-aggregated or summarized data to support analytical queries efficiently. While ER models can represent relationships between entities, they may not capture the need for pre-computed aggregates that are essential for reporting and analysis in a Data Warehouse.

Because of these limitations, Data Warehouses typically use specialized modeling techniques and architectures, such as dimensional modeling, which are better suited for the specific requirements of analytical querying and reporting on large volumes of data.



Steps to Create Dimensional Data Modeling

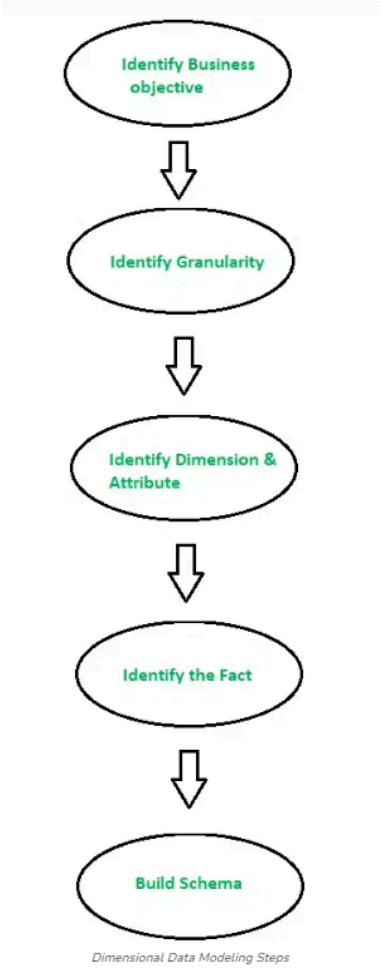
Step-1: Identifying the business objective: The first step is to identify the business objective. Sales, HR, Marketing, etc. are some examples of the need of the organization. Since it is the most important step of Data Modelling the selection of business objectives also depends on the quality of data available for that process.

Step 2: Identifying Granularity: Granularity is the lowest level of information stored in the table. The level of detail for business problems and its solution is described by Grain.

Step 3: Identifying Dimensions and their Attributes: Dimensions are objects or things. Dimensions categorize and describe data warehouse facts and measures in a way that supports meaningful answers to business questions. A data warehouse organizes descriptive attributes as columns in dimension tables. For Example, the data dimension may contain data like a year, month, and weekday.

Step-4: Identifying the Fact: The measurable data is held by the fact table. Most of the fact table rows are numerical values like price or cost per unit, etc.

Step-5: Building of Schema: We implement the Dimension Model in this step. A schema is a database structure. There are two popular schemes: Star Schema and Snowflake Schema.



Dimensional Data Modeling Steps

4. Define dimension, fact , fact table and dimension table with example.

- Facts are the measurements/metrics from your business process.
- For a Sales business process, a measurement would be a quarterly sales number

Dimension

- A category of information. For example, the time dimension.
- In simple terms, they give who, what, and where of a fact.
- E.g.In the Sales business process, for the fact quarterly sales number, dimensions would be
- Who - Customer Names
- Where - Location
- What - Product Name
- When - Time Dimension
- In other words, a dimension is a window to view information in the facts.

Attributes

- The Attributes are the various characteristics of the dimension
- E.g. In the Location dimension, the attributes can be State, Country, Zipcode, etc.
- Attributes are used to search, filter, or classify facts. Dimension Tables contain Attributes.

Fact Table

- A fact table is a primary table in dimension modeling.
- A fact table consists of the measurements, metrics, or facts of a business process.
- Eg. Monthly sales volume, Average Customer Balance, etc...
- A Fact Table contains
- 1. Measurements/facts
- 2. Foreign key to the dimension table

Dimension Table

- A dimension table contains the dimensions of a fact.
- They are joined to the fact table via a foreign key.
- Dimension tables are denormalized tables.
- The Dimension Attributes are the various columns in a dimension table
- Dimensions offer descriptive characteristics of the facts with the help of their attributes
- No limit is set for a number of dimensions
- The dimension can also contain one or more hierarchical relationships

5. Difference between star and snowflake schema.

Star Schema	Snowflake Schema
Hierarchies for the dimensions are stored in the dimensional table.	Hierarchies are divided into separate tables.
It contains a fact table surrounded by dimension tables.	One fact table surrounded by dimension table which are in turn surrounded by dimension table
In a star schema, only single join creates the relationship between the fact table and any dimension tables.	A snowflake schema requires many joins to fetch the data.
Simple DB Design.	Very Complex DB Design.
Denormalized Data structure and query also run faster.	Normalized Data Structure.
High level of Data redundancy	Very low-level data redundancy
Single Dimension table contains aggregated data.	Data Split into different Dimension Tables.
Cube processing is faster.	Cube processing might be slow because of the complex join.
Offers higher performing queries using Star Join Query Optimization. Tables may be connected with multiple dimensions.	The Snowflake schema is represented by centralized fact table which is unlikely connected with multiple dimensions.

6. Design star and snowflake schema for given system.

7. Difference between OLTP and OLAP.

OLTP	OLAP
OLTP is an online transactional system.	OLAP is an online analysis and data retrieving process.
It is characterized by large numbers of short online transactions.	It is characterized by a large volume of data.
OLTP is an online database modifying system.	OLAP is an online database query management system.
OLTP uses traditional DBMS.	OLAP uses the data warehouse.
Insert, Update, and Delete information from the database.	Mostly select operations
OLTP and its transactions are the sources of data.	Different OLTP databases become the source of data for OLAP.
OLTP database must maintain data integrity constraints.	OLAP database does not get frequently modified. Hence, data integrity is not an issue.
It's response time is in a millisecond.	Response time in seconds to minutes.
The data in the OLTP database is always detailed and organized.	The data in the OLAP process might not be organized.
Allow read/write operations.	Only read and rarely write.
It is a customer-oriented process.	It is a market oriented process.
Queries in this process are standardized and simple.	Complex queries involving aggregations.
Complete backup of the data combined with incremental backups.	OLAP only need a backup from time to time. Backup is not important compared to OLTP
DB design is an application-oriented example: Database design changes with the industry like retail, airline, banking, etc.	DB design is subject-oriented. Example: Database design changes with subjects like sales, marketing, purchasing, etc.

OLTP	OLAP
It is used by Data critical users like clerk, DBA & Data Base professionals.	It is used by Data knowledge users like workers, managers, and CEO.
It is designed for real time business operations.	It is designed for analysis of business measures by category and attributes.
Transaction throughput is the performance metric	Query throughput is the performance metric.
This kind of Database allows thousands of users.	This kind of Database allows only hundreds of users.
It helps to Increase user's self-service and productivity	Help to Increase the productivity of business analysts.
It provides a fast result for daily used data.	It ensures that response to the query is quicker consistently.
It is easy to create and maintain.	It lets the user create a view with the help of a spreadsheet.

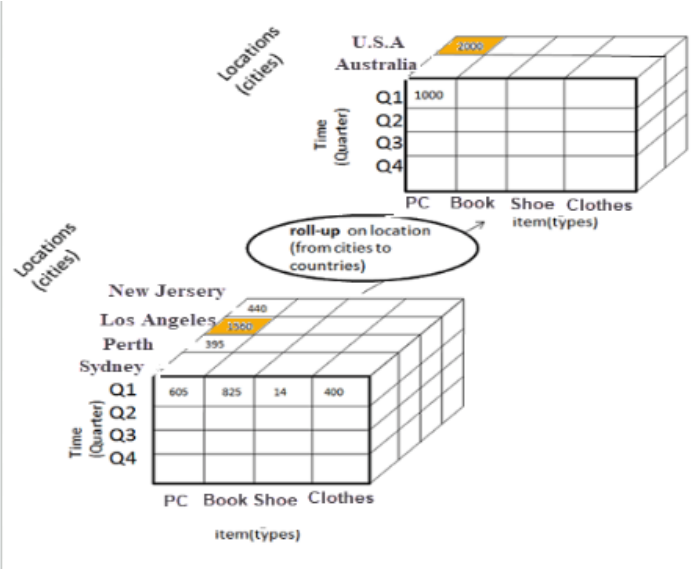
8. What are different OLAP operations?Explain with example.

Roll-up:

Roll-up is also known as "consolidation" or "aggregation." The Roll-up operation can be performed in 2 ways

- Reducing dimensions
- Climbing up concept hierarchy.

In the roll-up process, at least one or more dimensions need to be removed.

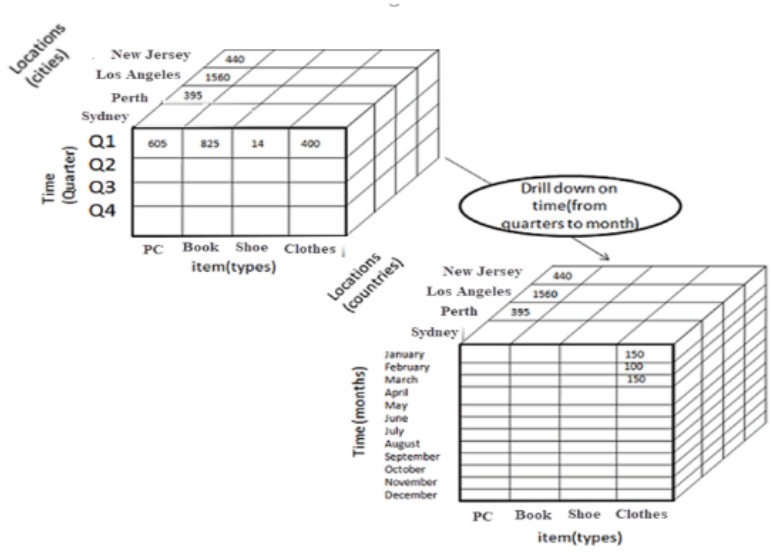


In this example, cities New Jersey and Los Angeles are rolled up into country USA
The sales figures of New Jersey and Los Angeles are 440 and 1560 respectively. They become 2000 after roll-up
In this aggregation process, data in location hierarchy moves up from the city to the country.
In this example, the Cities dimension is removed.

Drill-down

In drill-down, data is fragmented into smaller parts. It is the opposite of the rollup process. It can be done via

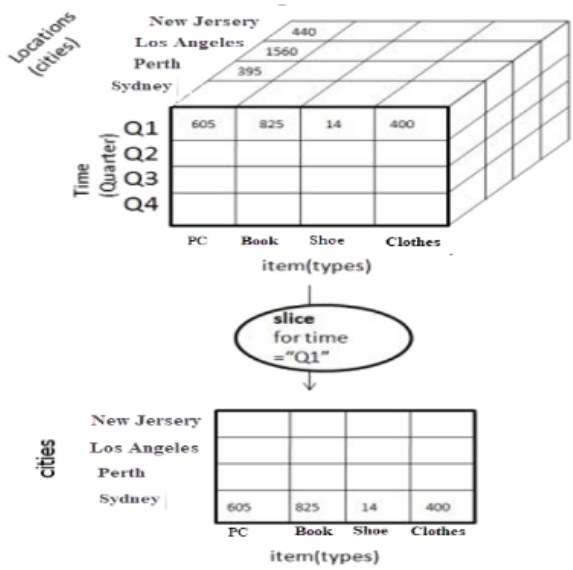
- Moving down the concept hierarchy
- Increasing a dimension



In this Example,
Quarter Q1 is drilled down to months
Here dimension Months is added.

Slice

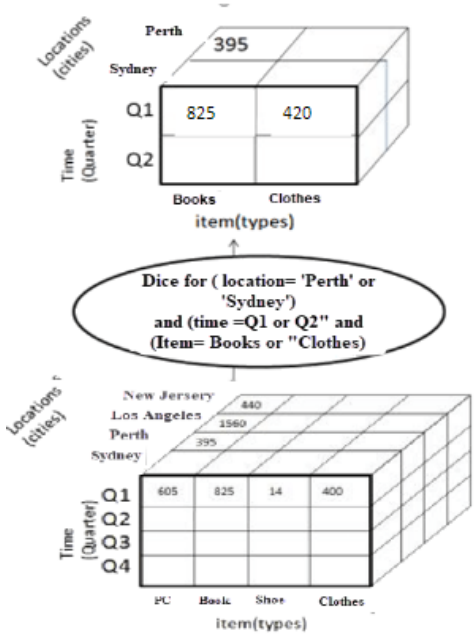
Here, one dimension is selected, and a new sub-cube is created.



In this example,
Dimension Time is sliced with Q1 as the filter.
A new cube is created altogether.

Dice

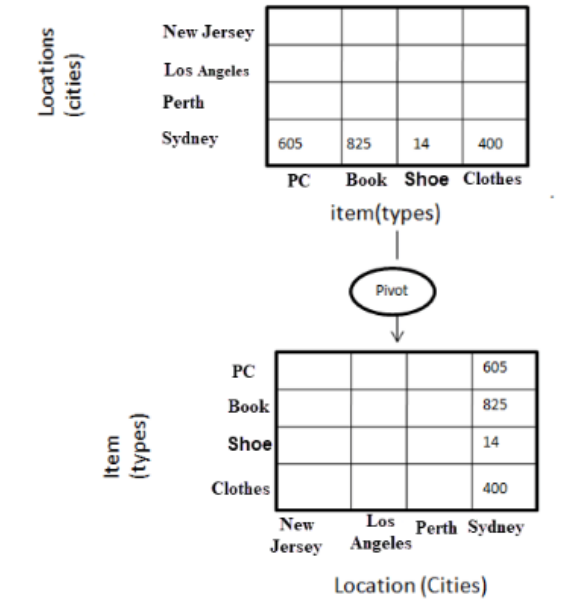
This operation is similar to a slice. The difference in dice is you select 2 or more dimensions that result in the creation of a sub-cube.



Dice operation in OLAP

Pivot

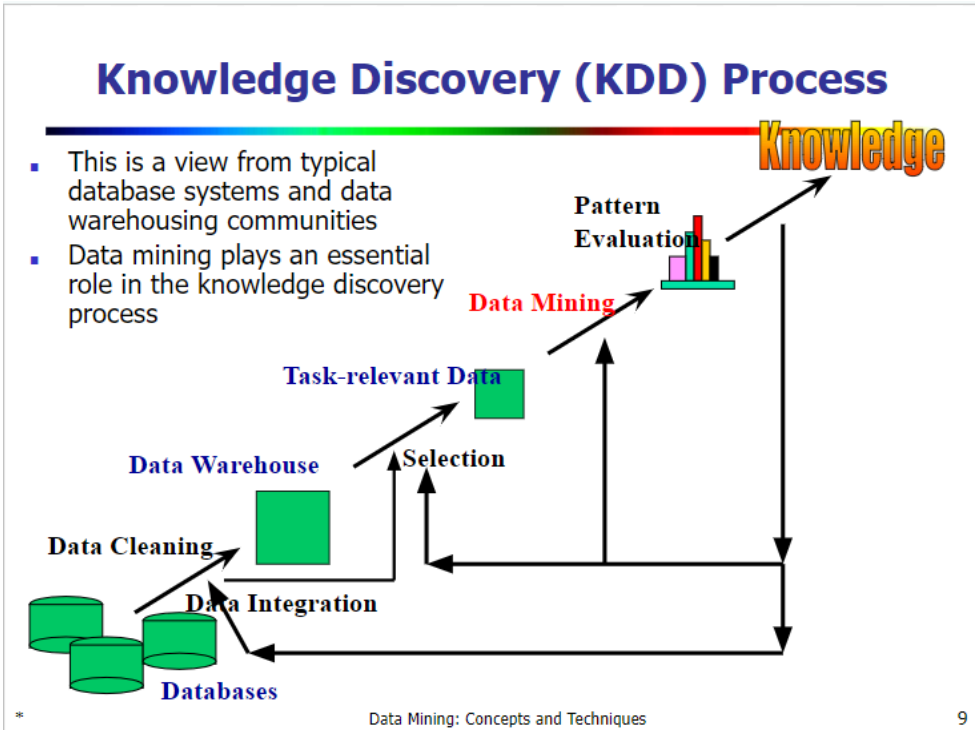
In Pivot, you rotate the data axes to provide a substitute presentation of data.



Pivot operation in OLAP
Here the pivot is based on item types

9. Problems on writing a sequence of OLAP operations for the given query.

10. Explain steps of KDD



KDD Process

KDD (Knowledge Discovery in Databases) is a process that involves the extraction of useful, previously unknown, and potentially valuable information from large datasets. The KDD process is an iterative process and it requires multiple iterations of the above steps to extract accurate knowledge from the data.The following steps are included in **KDD process:**

Data Cleaning
Data cleaning is defined as removal of noisy and irrelevant data from collection.

1. Cleaning in case of Missing values.
2. Cleaning noisy data, where noise is a random or variance error.
3. Cleaning with Data discrepancy detection and Data transformation tools.

Data Integration
Data integration is defined as heterogeneous data from multiple sources combined in a common source(DataWarehouse). Data integration using Data Migration tools, Data Synchronization tools and ETL(Extract-Load-Transformation) process.

Data Selection
Data selection is defined as the process where data relevant to the analysis is decided and retrieved from the data collection. For this we can use Neural network, Decision Trees, Naive bayes, Clustering, and Regression methods.

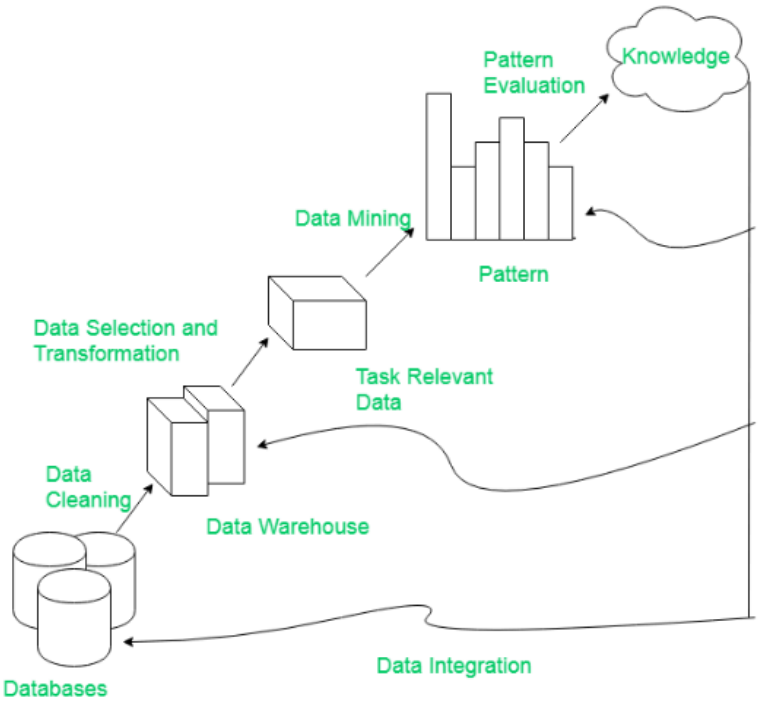
Data Transformation
Data Transformation is defined as the process of transforming data into appropriate form required by mining procedure. Data Transformation is a two step process:

1. Data Mapping: Assigning elements from source base to destination to capture transformations.
2. Code generation: Creation of the actual transformation program.

Data Mining
Data mining is defined as techniques that are applied to extract patterns potentially useful. It transforms task relevant data into patterns, and decides purpose of model using classification or characterization.

Pattern Evaluation
Pattern Evaluation is defined as identifying strictly increasing patterns representing knowledge based on given measures. It find interestingness score of each pattern, and uses summarization and Visualization to make data understandable by user.

Knowledge Representation
This involves presenting the results in a way that is meaningful and can be used to make decisions.



Note: KDD is an iterative process where evaluation measures can be enhanced, mining can be refined, and new data can be integrated and transformed in order to get different and more appropriate results. Preprocessing of databases consists of Data cleaning and Data Integration.

11. State any 2 decision making activities for which organizations are using data in DWH.

Many organizations use this information to support business decision-making activities, including

(1) increasing customer focus, which includes the analysis of customer buying patterns ;

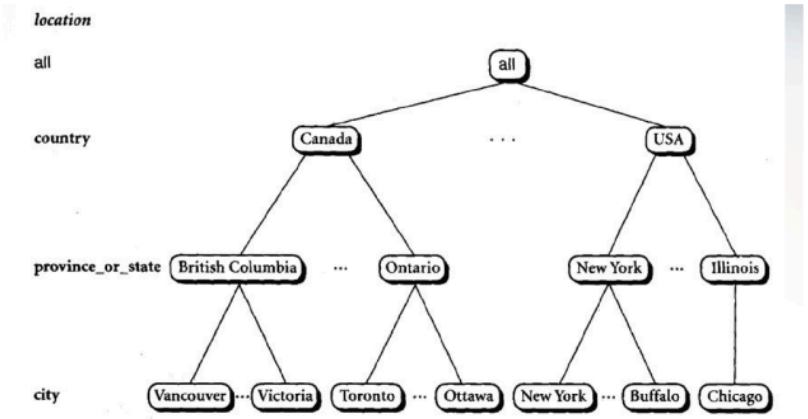
(2) repositioning products and managing product portfolios (by comparing the performance of sales by quarter, by year, and by geographic regions in order to fine-tune production strategies);

(3) analyzing operations and looking for sources of profit; and

(4) managing customer relationships, making environmental corrections, and managing the cost of corporate assets.

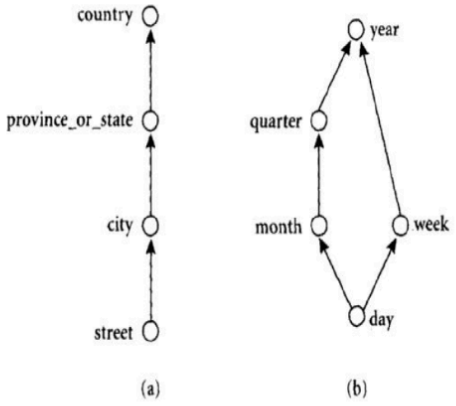
12. What is concept hierarchy, partial and total order concept hierarchy? Explain with an example.

It is a sequence of mappings from a set of low-level concepts to higher-level, more general concepts



Concept Hierarchy

- A Concept Hierarchy may also be a total order or partial order among attributes in a database schema
- It may also be defined by discretizing or grouping values for a given dimension or attribute, resulting in a **set-grouping** hierarchy
- Concept Hierarchies may be provided manually by
  - System users
  - Domain Experts
  - Knowledge Engineers
  - Automated Statistical Analysis



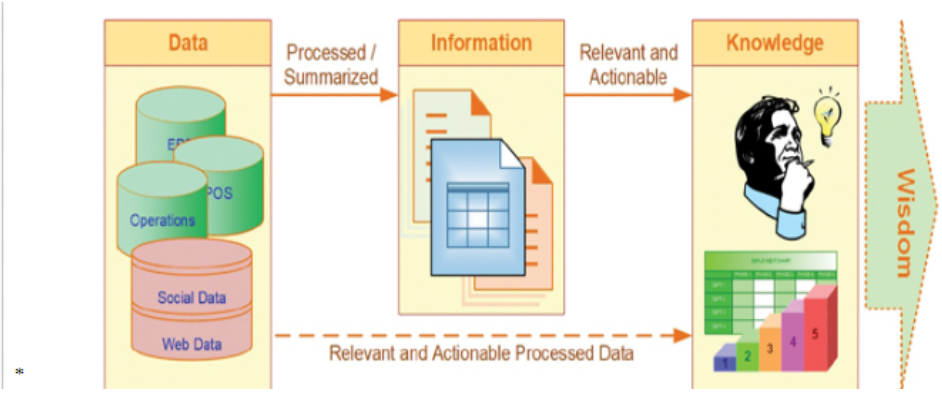
13. What is data mining? State applications of data mining.

Data mining is the process of converting data into information and then into knowledge.

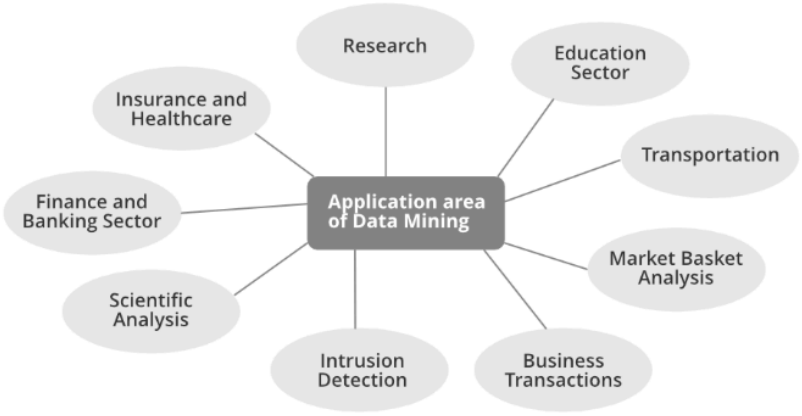
Knowledge is very distinct from data and information

Knowledge is information that is contextual, relevant, and actionable.

knowledge has strong experiential and reflective elements that distinguish it from information in a given context.



Data mining is a process that involves using statistical, mathematical, and artificial intelligence techniques and algorithms to extract and identify useful information and subsequent knowledge (or patterns) from large sets of data.



Marketing and CRM:
-To identify most likely buyers of new products
-To identify root causes of customer attrition so as to improve customer retention
-To discover time variant associations between products and services to maximize sale and find most profitable customers.
Banking and Finance:
-To detect fraudulent credit card and online banking transactions
-To optimize the cash return by forecasting cash flow on banking entities
-To streamline and automate the processing of loan applications by accurately predicting the most probable defaulters.
-To maximize customer value by identifying and selling the products and services that customers are most likely to buy.
Retailing and Logistics:
-To identify accurate sales volume at specific retail locations in order to determine correct inventory levels.
-To do an MBA to improve store layout and optimize sales promotions
-To forecast consumption levels for different product types.
-To discover interesting patterns in the movement of products in a supply chain by analyzing sensory and RFID data.
Manufacturing:
- To predict machine failures using sensory data
- To discover novel patterns to identify and improve product quality.
Brokerages and Security Tradings:
-To predict when and how much certain stock/bond prices will change.
-To forecast the range of market fluctuations, and direction of fluctuations
-To assess the effect of particular issues/events on market movements.
-To identify and prevent fraudulent activities in security trading.
Insurance:
- To predict which customers will buy new policies
- Identify fraudulent behavior of customers

- Prevent incorrect claim payments

Computer Hardware and Software:

-To predict disk failure

- To identify and filter unwanted web contents and email messages

-To identify potentially unsecured software products

Government and Defense:

-To forecast the cost of moving military personnel and equipment.

-To predict resource consumption for better planning and budgeting

Travel and Lodging:

- To predict sales of different services to optimally price these services.
- To forecast demand at different locations to better allocate limited organizational resources. .
- To identify the most profitable customers and provide them with personalized services.
- To retain valuable employees by identifying and acting on the root causes for attrition

Health and Healthcare:

- To identify successful medical therapies for different illnesses.
- To identify people without health insurance and the reasons behind it.
- To forecast the time of demand at different service locations to optimally allocate organizational resources.
- To retain valuable employees by identifying root causes for attrition

Entertainment:

To analyze viewer data to determine which programs to show during prime time.

To decide where to insert advertisements so as to maximize the returns.

To predict the financial success of the movies before they are produced.

Sports:

To improve the performance of NBA teams in the US

To increase the chances of winning.

14. What are the different types of patterns that can be mined?

Data mining functionalities are used to specify the kinds of patterns to be found in data mining tasks.

Descriptive mining tasks: Deals with the General characteristics and converts them into relevant and useful information

Predictive mining tasks: Predicts future values by analyzing data patterns and their outcomes based on past data.

Descriptive DM Functionalities

1. Class/Concept Description:

Data entries can be associated with the classes or concepts.

These descriptions can be derived using

(1) data characterization, by summarizing the data of the class under study (often called the target class) in general terms,

Example: At an electronic store a Customer relationship manager asks to Summarize the characteristics of customers who spend more than Rs.10000 a year at the store.

or

(2) data discrimination, by comparison of the target class with one or a set of comparative classes (often called the contrasting classes),

Example: A customer relationship manager at an Electronics store may want to compare two groups of customers—those who shop for computer products regularly (e.g., more than twice a month) and those who rarely shop for such products (e.g. less than three times a year).

or

(3) both data characterization and discrimination.

Mining of frequent patterns:

Patterns that occur frequently in data.

It includes--

- **Frequent item**: refers to a set of items that often appear together in a transactional data set
- **Frequent subsequences** (also known as sequential patterns): A frequently occurring subsequence like laptop digital camera memory card
- **Frequent substructures:**
- A substructure can refer to different structural forms (e.g., graphs, trees, or lattices) that may be combined with itemsets or subsequences.
- If a substructure occurs frequently, it is called a (frequent) structured pattern.

Mining frequent patterns leads to the discovery of interesting associations and correlations within data.

Association Analysis

Defines relationships between the data and predefined association rules.

Suppose that, as a marketing manager at an Electronics store, you want to know which items are frequently purchased together

A rule, mined from the Electronics store transactional database

Association rules that contain a single predicate are referred to as single-dimensional association rules.

Suppose, instead, that we are given the Electronics relational database related to purchases. A data mining system may find association rules like

Association rules that contain more than one predicate/attribute are referred to as multi-dimensional association rules.

Clustering:

- can be used to generate class labels for a group of data. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the interclass similarity.

- i.e. clusters of objects are formed so that objects within a cluster have high similarity, but are rather dissimilar to objects in other clusters.
- Clustering can also facilitate taxonomy formation Organization of observations into a hierarchy of classes that group similar events together.

Example: Cluster analysis can be performed on Electronics store customer data to identify homogeneous subpopulations of customers. These clusters may represent individual target groups for marketing

Predictive Data mining functionalities

Predicts future values by analyzing data patterns and their outcomes based on past data.

- Classification
- Regression
- Outlier analysis

Classification:

- Is the process of finding a model (or function) that describes and distinguishes data classes or concepts.
- The models are derived based on the analysis of a set of training data (i.e., data objects for which the class labels are known).
- The model is used to predict the class label of objects for which the class label is unknown.
- The derived model may be represented in various forms, such as classification rules (i.e., IF-THEN rules), decision trees, mathematical formulae, or neural networks

Regression:

- Whereas classification predicts categorical (discrete, unordered) labels, regression models continuous-valued functions. That is, regression is used to predict missing or unavailable numerical data values rather than (discrete) class labels.
- The term prediction refers to both numeric prediction and class label prediction.
- Regression analysis is a statistical methodology that is most often used for numeric prediction.
- Regression also encompasses the identification of distribution trends based on the available data.

Outlier Analysis:

- Outlier: A data object that does not comply with the general behavior of the data
- Many data mining methods discard outliers as noise or exceptions. However, in some applications (e.g., fraud detection), the rare events can be more interesting than the more regularly occurring ones.
- Outliers may be detected using statistical tests that assume a distribution or probability model for the data, or using distance measures where objects that are remote from any other cluster are considered outliers.

- Example: Outlier analysis may uncover fraudulent usage of credit cards by detecting purchases of unusually large amounts for a given account number in comparison to regular charges incurred by the same account.
- Outlier values may also be detected with respect to the locations and types of purchase, or the purchase frequency.
- It is used in observing the change in trends of buying patterns of a customer.

15. Draw a 3 tier data warehousing architecture.

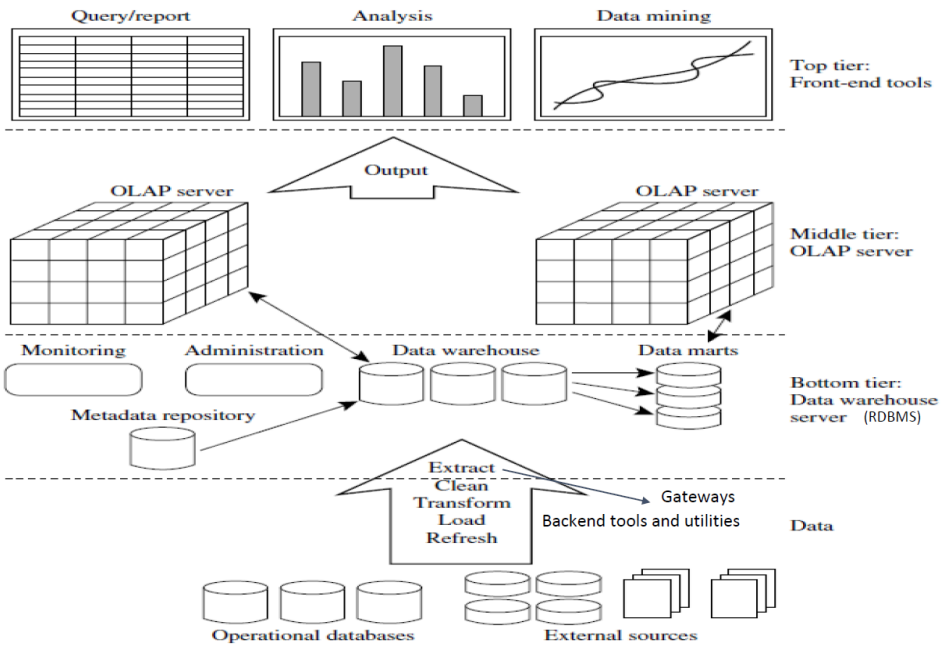


Figure 4.1 A three-tier data warehousing architecture.

Module 2:

1. What are the different types of attributes? Explain with examples

Attribute (or dimensions, features, variables):
A data field, representing a characteristic or feature of a data object.
e.g., customer _ID, name, address

Categorical/Qualitative Attribute

1. Nominal:

- categories, states, or "names of things"
  - Hair_color = {auburn, black, blond, brown, grey, red, white}
  - marital status, occupation, ID numbers, zip codes
- In the cases of nominal attributes with numeric values e.g. Cust_ID, the numbers are not intended to be used quantitatively.
- Also in the case of numeric nominal attributes, values do not have any meaningful order about them.

2. Binary attributes

- Nominal attribute with only 2 categories/states (0 or 1)
    - 0: attribute is absent
    - 1: attribute is present
- Symmetric binary: both outcomes equally important
    - e.g., gender
- Asymmetric binary: outcomes not equally important.
    - e.g., medical test (positive vs. negative)
    - Convention: assign 1 to the most important outcome (e.g., HIV positive)
- If two states are True and False,then called as Boolean Attribute

3. Ordinal Attributes:

- Values have a meaningful order or a ranking among them but the magnitude between successive values is not known.
    - Ex: Size = {small, medium, large}, grades, professor rankings
- Useful for registering subjective assessments of qualities that cannot be measured objectively; thus often used in surveys for ratings.
    - E.g. Customer satisfaction survey
- We can compute mean and median but not mode for the ordinal attributes.
- Note: nominal, binary, and ordinal attributes are qualitative attributes.

Numeric Attributes /Quantitative Attributes

Represent measurable quantity in integer or real values

1. Interval scaled attributes

- Measured on a scale of equal-sized units
- Values have order and can be positive, 0, or negative
    - E.g., the temperature in C˚or F˚, calendar dates
- We can obtain a ranking of objects by ordering the values.
- Also allow us to compare and quantify the difference between values.
- No true zero-point -We can not speak of values in terms of ratio.
    - e.g. without a true zero point, we can't say that 10 C˚ is twice as warm as 5C˚.
- Mean , Median and Mode

2. Ratio scaled attributes

- Inherent zero-point
- The values are ordered, and we can also compute the difference between values, as well as the mean, median, and mode
- Examples: Count attributes such as years of experience and number of words attribute for a document
- attributes to measure age, weight, height and monetary quantities (e.g., you are 100 times richer with $100 than with $1).

Discrete vs. Continuous Attributes

Discrete Attribute

- Has only a finite or countably infinite set of values which may or may not be represented as integers.
    - E.g., zip codes, profession, or the set of words in a collection of documents
- Sometimes, represented as integer variables

- Note: Binary attributes are a special case of discrete attributes

Continuous Attribute

- Has real numbers as attribute values
    - E.g., temperature, height, or weight
- Practically, real values can only be measured and represented using a finite number of digits
- Continuous attributes are typically represented as floating-point variables

2. Problems on basic statistical descriptions of data like finding mean, median, midrange standard deviation, variance,modes for given data.Drawing q-q plot and boxplot for given data.

Mean (algebraic measure) (sample vs. population):

The most common and effective numeric measure of the "center" of a set of data is the (arithmetic) mean.

Let x1, x2, x3, x4, ….. xN be a set of N values or observations, such as for some numeric attribute X, like salary.

The mean of this set of values is

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}.$$

$$\bar{x} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i}$$

Weighted mean is :

Problem with mean is its sensitivity to extreme values.

For skewed (asymmetric) data, a better measure of the center of data is the median

3. What is a five number summary of data?

- Five-number summary of a distribution (Minimum, Q1, Median, Q3, Maximum)
- It is more informative to also provide the two quartiles Q1 and Q3, along with the median
- A common rule of thumb for identifying suspected outliers is to single out values falling at least 1.5IQR above the third quartile or below the first quartile.
- Because Q1, the median, and Q3 together contain no information about the endpoints (e.g., tails) of the data, a fuller summary of the shape of a distribution can be obtained by providing the lowest and highest data values as well. This is known as the five-number summary.
- Upper Limit / Maximum = Q3 + 1.5*IQR
- Lower Limit / Minimum = Q1 - 1.5*IQR
- The five-number summary of distribution consists of the median (Q2), the quartiles Q1 and Q3, and the smallest and largest individual observations, written in the order of Minimum, Q1, Median, Q3, and Maximum.

4. How can we compute dissimilarity between two binary attributes?

how can we compute the dissimilarity between two binary attributes?

One approach involves computing a dissimilarity matrix from the given binary data. If all binary attributes are thought of as having the same weight, we have the 2 x 2 contingency table as shown below

Object *i* \ Object *j*	1	0	sum
1	q	r	$q+r$
0	s	t	$s+t$
sum	$q+s$	$r+t$	p

Where q = the number of attributes that equal 1 for both objects *i* and *j*,

r = the number of attributes that equal 1 for object *i* but equal 0 for object *j*,

s = the number of attributes that equal 0 for object *i* but equal 1 for object *j*,

t = the number of attributes that equal 0 for both objects *i* and *j*.

p = The total number of attributes = $q + r + s + t$.

35

Proximity Measure for Binary Attributes

A contingency table for binary data

Object *i* \ Object *j*	1	0	sum
1	q	r	$q+r$
0	s	t	$s+t$
sum	$q+s$	$r+t$	p

Distance measure for symmetric binary variables:

$$d(i,j) = \frac{r+s}{q+r+s+t}$$

Distance measure for asymmetric binary variables:

$$d(i,j) = \frac{r+s}{q+r+s}$$

Jaccard coefficient (similarity measure for asymmetric binary variables):

$$sim_{Jaccard}(i,j) = \frac{q}{q+r+s}$$

= 1- d(i,j)

36

Dissimilarity between Binary Attributes

Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- *Gender is a symmetric attribute*
- *The remaining attributes are asymmetric binary*
- *Let the values Y and P be 1, and the value N be 0*

Suppose distance is computed based on only asymmetric attributes

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

$$d(i,j) = \frac{r+s}{q+r+s}$$

- These measurements suggest that Jim and Mary are unlikely to have a similar disease because they have the highest dissimilarity value among the three pairs.

Of the three patients, Jack and Mary are most likely to have a similar disease.

37

5. What is Euclidean distance, Manhattan distance, Minkowski distance? Problems on computing these distances between given objects.

Dissimilarity of Numeric Data:

- Euclidean distance: The most popular distance measure

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two objects described by numeric attributes.

- Manhattan (or city block) distance: named so because it is the distance in blocks between any two points in a city (such as 2 blocks down and 3 blocks over for a total of 5 blocks).
  - The distance between two points measured along axes at right angles

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|.$$

Dissimilarity of Numeric Data: Minkowski Distance

The Euclidean and the Manhattan distance satisfy the following mathematical properties:

- **Non-negativity:** $d(i, j) >= 0$: Distance is a non-negative number.
- **Identity of indiscernibles:** $d(i, j) = 0$:The distance of an object to itself is 0.
- **Symmetry:** $d(i, j) = d(j, i)$: Distance is a symmetric function.
- **Triangle inequality:** $d(i, j) <= d(i, k) + d(k, j)$: Going directly from object *i* to object *j* in space is no more than making a detour over any other object *k*.

A measure that satisfies these conditions is known as **metric**.

Dissimilarity of Numeric Data: Minkowski Distance

Minkowski distance: A generalization of Euclidean and Manhattan distances

$$d(i, j) = \sqrt[h]{|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \cdots + |x_{ip} - x_{jp}|^h}$$

where i = (x_{i1}, x_{i2}, …, x_{ip}) and j = (x_{j1}, x_{j2}, …, x_{jp}) are two objects described by p numeric attributes and h is a real number such that h >= 1.

- Also called as L_p norm where p refers to h.

Special Cases of Minkowski Distance

- *h = 1: Manhattan (city block, L_1 norm) distance*

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

- *h = 2: (L_2 norm) Euclidean distance*

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

- *$h \to \infty$. "supremum" (L_{max} norm, L_∞ norm, Chebyshev distance) distance.*

To compute it, we find the attribute *f* that gives the maximum difference in values between the two objects.

This difference is the supremum distance, defined more formally as:

$$d(i, j) = \lim_{h\to\infty} \left(\sum_{f=1}^{p} |x_{if} - x_{jf}|^h \right)^{\frac{1}{h}} = \max_{f}^{p} |x_{if} - x_{jf}|$$

Example: **Supremum distance.** Let's use the two objects, $\boldsymbol{x_1} = (1, 2)$ and $\boldsymbol{x_2} = (3, 5)$ The second attribute gives the greatest difference between values for the objects, which is $5 - 2 = 3$. This is the supremum distance between both objects.

6. What is cosine similarity?problems on finding similarity between given documents.

Cosine Similarity

- A **document** can be represented by thousands of attributes, each recording the frequency of a particular word (such as keywords) or phrase in the document.

Document	*team*	*coach*	*hockey*	*baseball*	*soccer*	*penalty*	*score*	*win*	*loss*	*season*
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

- Applications: information retrieval, biologic taxonomy, gene feature mapping, ...
- Cosine measure: If x and y are two vectors (e.g., term-frequency vectors), then

  cos(x, y) = (x · y) /||x|| ||y|| ,

  where · indicates vector dot product, ||x||: the Euclidean norm of vector x, defined as

  $\sqrt{x_1^2 + x_2^2 + \cdots + x_p^2}$. **= length of vector x**

- The Cosine measure computes the cosine of the angle between vectors x and y.
- A cosine value of 0 means that the two vectors are at 90 degrees to each other (orthogonal) and have no match.
- The closer the cosine value to 1, the smaller the angle and the greater the match between vectors.

Example: Cosine Similarity

cos(x, y) = = (x · y) / ||x|| ||y|| ,
where · indicates vector dot product, ||d|: the length of vector d

Ex: Find the **similarity** between documents 1 and 2.

x = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0)
y = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1)

x·y = 5*3+0*0+3*2+0*0+2*1+0*1+0*1+2*1+0*0+0*1 = 25

$||x|| = (5*5+0*0+3*3+0*0+2*2+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$

$||y|| = (3*3+0*0+2*2+0*0+1*1+1*1+0*0+1*1+0*0+1*1)^{0.5} = (17)^{0.5} = 4.12$

cos(x, y) = 0.94 -----Quite similar

7. Problems based on finding dissimilarity matrices between nominal,binary and ordinal attributes .

8. Explain in brief the major tasks in data preprocessing.

- Data cleaning
  - Fill in missing values, smooth out the noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
  - Integration of multiple databases, data cubes, or files
- Data transformation
  - Normalization and aggregation
- Data reduction
  - Obtains reduced representation in volume but produces the same or similar analytical results
- Data discretization
  - Part of data reduction but with particular importance, especially for numerical data

9. What are the different ways to handle missing data?

Data Cleaning

- Data cleaning tasks
  - Fill in missing values
  - Identify outliers and smooth out noisy data
  - Correct inconsistent data
  - Resolve redundancy caused by data integration

Data Cleaning:Missing Data

- Data is not always available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
  - equipment malfunction
  - inconsistent with other recorded data and thus deleted
  - data not entered due to misunderstanding
  - certain data may not be considered important at the time of entry
- Missing data may need to be inferred.

How to Handle Missing Data?

1. **Ignore the tuple:** usually done when class label is missing assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably.

1. **Fill in the missing value manually:** tedious + infeasible?

1. **Use a global constant to fill in the missing value**: Replace all missing attribute values by the same constant such as a label like *"Unknown"*

1. **Use a measure of central tendency for the attribute (e.g., the mean or median) to fill in the missing value**.

1. **Use the attribute mean or median for all samples belonging to the same class as the given tuple.**

1. **Use The most probable value: inference-based such as Bayesian formula or decision tree**

10. What are the different ways to handle noisy data?
AND 16. Binning different types and problems bases on binning
And 17. What is noise? Explain data smoothing methods as noise removal technique to divide given data into bins of size 3
AND 18. Noise removal techniques

Data Cleaning:Noisy Data

- Noise: random error or variance in a measured variable
- Meaningless data that can not be interpreted by machines
- Noisy data (incorrect values) may come from
  - Faulty data collection instruments
  - Human or computer error at data entry
  - Errors in data transmission

How to Handle Noisy Data?

I. Binning

- Binning is a technique where we sort the data and then partition the data into equal frequency bins. Then you may either replace the noisy data with the bin mean bin median or the bin boundary.

- There are three methods for smoothing data in the bin.
  - **Smoothing by bin mean method:** In this method, the values in the bin are replaced by the mean value of the bin.
  - **Smoothing by bin median:** In this method, the values in the bin are replaced by the median value.
  - **Smoothing by bin boundary:** In this method, the minimum and maximum values of the bin values are taken, and the closest boundary value replaces the values.

I. Binning Example

- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

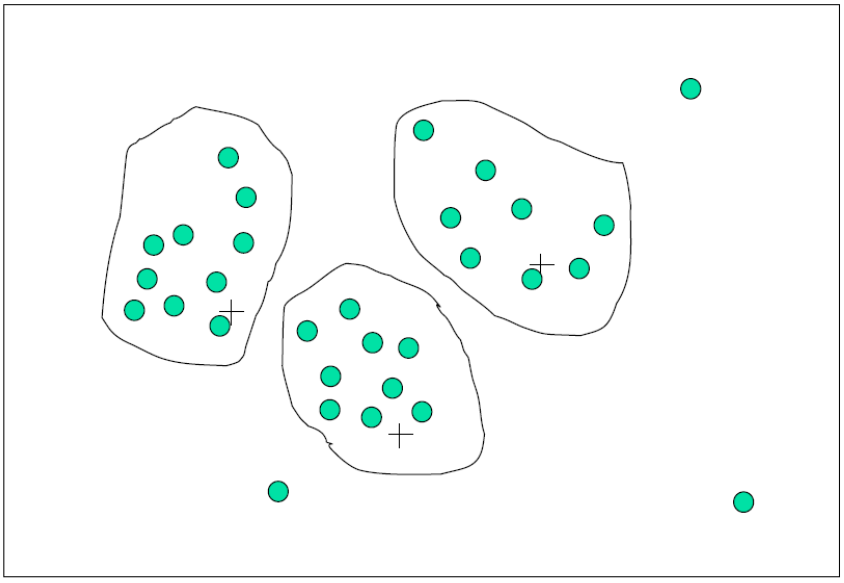
* Partition into equal-frequency (equi-depth) bins:
  - Bin 1: 4, 8, 9, 15
  - Bin 2: 21, 21, 24, 25
  - Bin 3: 26, 28, 29, 34
* Smoothing by bin means:
  - Bin 1: 9, 9, 9, 9
  - Bin 2: 23, 23, 23, 23
  - Bin 3: 29, 29, 29, 29
* Smoothing by bin boundaries:
  - Bin 1: 4, 4, 4, 15
  - Bin 2: 21, 21, 25, 25
  - Bin 3: 26, 26, 26, 34

II. Regression

- This is used to smooth the data and will help to handle data when unnecessary data is present.
- For the analysis, purpose regression helps to decide the variable which is suitable for our analysis.
  - **Linear regression** refers to finding the best line to fit between two variables so that one can be used to predict the other.
  - **Multiple linear regression** involves more than two variables.
  - Using regression to find a mathematical equation to fit into the data helps to smooth out the noise.

III. Outlier Analysis

- detect and remove outliers



11. Problems on correlation analysis for categorical(Chi square test) and numerical data./ Problems based on finding correlation between attributes (Chi Square test, Pearson correlation coefficient, covariance, etc…)

12. What are the different data transformation strategies?
AND
13. Problems on min max ,z score and decimal scaling normalization.
AND
15. Data transformation techniques

Data Transformation

- Data transformation is a technique used to **convert the raw data into a suitable format** that efficiently eases data mining and retrieves strategic information.
- Data transformation includes data cleaning techniques and a data reduction technique to convert the data into the appropriate form.
- Data transformation is an essential data preprocessing technique that must be performed on the data before data mining to provide patterns that are easier to understand.
- Data transformation changes the format, structure, or values of the data and converts them into clean, usable data.

1. Data Smoothing
2. Data Aggregation
3. Data Generalization: concept hierarchy climbing
4. Data Normalization: scaled to fall within a small, specified range
   - min-max normalization
   - z-score normalization
   - normalization by decimal scaling
5. Attribute/feature construction
   - New attributes constructed from the given ones
6. Discretization

1. Data Smoothing

- Data smoothing is a process that is used to remove noise from the dataset using some algorithms.
- It allows for highlighting important features present in the dataset. It helps in predicting the patterns.
- When collecting data, it can be manipulated to eliminate or reduce any variance or any other noise form.
- The concept behind data smoothing is that it will be able to identify simple changes to help predict different trends and patterns. This serves as a help to analysts or traders who need to look at a lot of data which can often be difficult to digest for finding patterns that they wouldn't see otherwise.

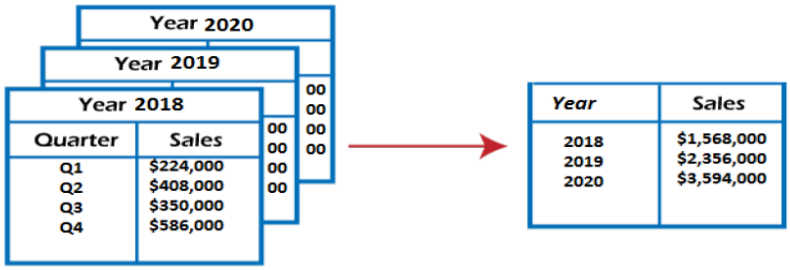
The noise is removed from the data using the techniques such as binning, regression, clustering.

- **Binning:** This method splits the sorted data into the number of bins and smoothens the data values in each bin considering the neighborhood values around it.
- **Regression:** This method identifies the relation among two dependent attributes so that if we have one attribute, it can be used to predict the other attribute.
- **Clustering:** This method groups similar data values and form a cluster. The values

2. Data Aggregation

- Data collection or aggregation is the method of storing and presenting data in a summary format.
- This is a crucial step since the accuracy of data analysis insights is highly dependent on the quantity and quality of the data used.
- Gathering accurate data of high quality and a large enough quantity is necessary to produce relevant results.

For example, we have a data set of sales reports of an enterprise that has quarterly sales of each year. We can aggregate the data to get the enterprise's annual sales report.



3. Data Generalization:concept hierarchy climbing

- It converts low-level data attributes to high-level data attributes using concept hierarchy.
- This conversion from a lower level to a higher conceptual level is useful to get a clearer picture of the data.

- Data generalization can be divided into two approaches:
  - Data cube process (OLAP) approach.
  - Attribute-oriented induction (AOI) approach.

For example, age data can be in the form of (20, 30) in a dataset. It is transformed into a higher conceptual level into a categorical value (young, old).

4. Normalization

(data scaled to fall within a small, specified range)

- Min-max normalization:

Min-max normalization performs a linear transformation on the original data. Suppose that min_A and max_A are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, v_i, of A to v_i' in the range $[new\_min_A, new\_max_A]$ by computing

$$v_i' = \frac{v_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A.$$

Min-max normalization preserves the relationships among the original data values. It will encounter an "out-of-bounds" error if a future input case for normalization falls outside of the original data range for A.

Example **Min-max normalization.** Suppose that the minimum and maximum values for the attribute *income* are \$12,000 and \$98,000, respectively. We would like to map *income* to the range $[0.0, 1.0]$. By min-max normalization, a value of \$73,600 for *income* is transformed to $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$.

Data Transformation by Normalization

- Z-score normalization

In **z-score normalization** (or *zero-mean normalization*), the values for an attribute, A, are normalized based on the mean (i.e., average) and standard deviation of A. A value, v_i, of A is normalized to v_i' by computing

$$v_i' = \frac{v_i - \bar{A}}{\sigma_A},$$

where $\bar{A}$ and σ_A are the mean and standard deviation, respectively, of attribute A.

$\bar{A} = \frac{1}{n}(v_1 + v_2 + \cdots + v_n)$ and σ_A is computed as the square root of the variance of A

- This method of normalization is useful when the actual minimum and maximum of attribute *A* are unknown, or when there are outliers that dominate the min-max normalization.

Example **z-score normalization.** Suppose that the mean and standard deviation of the values for the attribute *income* are \$54,000 and \$16,000, respectively. With z-score normalization, a value of \$73,600 for *income* is transformed to $\frac{73,600 - 54,000}{16,000} = 1.225$.

Data Transformation by Normalization

- Decimal scaling normalization:

Normalization by decimal scaling normalizes by moving the decimal point of values of attribute A. The number of decimal points moved depends on the maximum absolute value of A. A value, v_i, of A is normalized to v_i' by computing

$$v_i' = \frac{v_i}{10^j},$$

where j is the smallest integer such that $max(|v_i'|) < 1$.

Example **Decimal scaling.** Suppose that the recorded values of A range from -986 to 917. The maximum absolute value of A is 986. To normalize by decimal scaling, we therefore divide each value by 1000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917.

5. Attribute Construction

- In the attribute construction method, the new attributes consult the existing attributes to construct a new data set that eases data mining.

- New attributes are created and applied to assist the mining process from the given attributes. This simplifies the original data and makes the mining more efficient.

- For example, suppose we have a data set referring to measurements of different plots, i.e., we may have the height and width of each plot. So here, we can construct a new attribute 'area' from attributes 'height' and 'weight'.
- Attribute construction also helps understand the relations among the attributes in a data set.

6. Data Discretization

- This is a process of converting continuous data into a set of data intervals. Continuous attribute values are substituted by small interval labels.
- This makes the data easier to study and analyze.
- If a data mining task handles a continuous attribute, then its discrete values can be replaced by constant quality attributes. This improves the efficiency of the task.
- This method is also called a data reduction mechanism as it transforms a large dataset into a set of categorical data.
- Discretization also uses decision tree-based algorithms to produce short, compact, and accurate results when using discrete values.

For example, the values for the age attribute can be replaced by the interval labels such as (0-10, 11-20…) or (kid, youth, adult, senior).

14. State different data reduction strategies.

Better answer at : https://www.javatpoint.com/data-reduction-in-data-mining

Data Reduction Strategies

- Obtain a reduced representation of the data set that is much smaller in volume but yet produce the same (or almost the same) analytical results
- Sometimes, it is also performed to find the most suitable subset of attributes from a large number of attributes. This is known as dimensionality reduction.
- Data reduction also involves reducing the number of attribute values and/or the number of tuples.

1. **Data cube aggregation:** In this technique the data is reduced by applying OLAP operations like slice, dice or rollup. It uses the smallest level necessary to solve the problem.
2. **Dimensionality reduction:** The data attributes or dimensions are reduced. Not all attributes are required for data mining. The most suitable subset of attributes are selected by using techniques like forward selection, backward elimination, decision tree induction or a combination of forward selection and backward elimination.
3. **Data compression:** In this technique. large volumes of data is compressed i.e. the number of bits used to store data is reduced. This can be done by using lossy or lossless compression. In *loss compression,* the quality of data is compromised for more compression. In *lossless compression,* the quality of data is not compromised for higher compression level.
4. **Numerosity reduction :** This technique reduces the volume of data by choosing smaller forms for data representation. Numerosity reduction can be done using histograms, clustering or sampling of data. Numerosity reduction is necessary as processing the entire data set is expensive and time consuming.

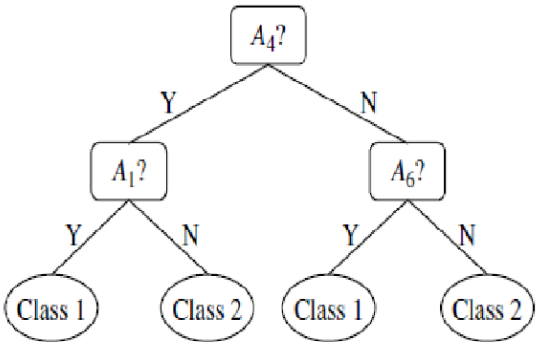
1.Dimensionality reduction:

- It eliminates outdated or redundant features.
- 3 methods are :
- **Wavelet Transform:**The **discrete wavelet transform (DWT)** is a linear signal processing technique that, when applied to a data vector ***X***, transforms it to a numerically different vector, ***X'*** of **wavelet coefficients**.
  The compressed data is obtained by retaining the smallest fragment of the strongest wavelet coefficients. Wavelet transform can be applied to data cubes, sparse data, or skewed data.

- **Principal Component Analysis:** Suppose we have a data set to be analyzed that has tuples with n attributes. The principal component analysis identifies k independent tuples with n attributes that can represent the data set.
  In this way, the original data can be cast on a much smaller space, and dimensionality reduction can be achieved. Principal component analysis can be applied to sparse and skewed data.

- **Attribute Subset Selection:** The attribute subset selection reduces the volume of data by eliminating redundant and irrelevant attributes.
  The most suitable subset of attributes are selected by using techniques like forward selection, backward elimination, decision tree induction or a combination of forward selection and backward elimination.
  The attribute subset selection ensures that we get a good subset of original attributes even after eliminating the unwanted attributes. The resulting probability of data distribution is as close as possible to the original data distribution using all the attributes.

Greedy(heuristic) methods for attribute subset selection

Forward selection	Backward elimination	Decision tree induction
Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ Initial reduced set: $\{\}$ $\Rightarrow \{A_1\}$ $\Rightarrow \{A_1, A_4\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_4, A_5, A_6\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$  => Reduced attribute set: $\{A_1, A_4, A_6\}$

1. Stepwise forward selection: The procedure starts with an empty set of attributes as the reduced set. The best of the original attributes is determined and added to the reduced set. At each subsequent iteration or step, the best of the remaining original attributes is added to the set.
2. Stepwise backward elimination: The procedure starts with the full set of attributes. At each step, it removes the worst attribute remaining in the set.
3. Combination of forward selection and backward elimination: The stepwise forward selection and backward elimination methods can be combined so that, at each step, the procedure selects the best attribute and removes the worst from among the remaining attributes.
4. Decision tree induction: Decision tree algorithms (e.g., ID3, C4.5, and CART) were originally intended for classification. Decision tree induction constructs a flowchart like structure where each internal (nonleaf) node denotes a test on an attribute, each branch corresponds to an outcome of the test, and each external (leaf) node denotes a class prediction. At each node, the algorithm chooses the "best" attribute to partition the data into individual classes.
When decision tree induction is used for attribute subset selection, a tree is constructed from the given data. All attributes that do not appear in the tree are assumed to be irrelevant. The set of attributes appearing in the tree form the reduced subset of attributes.
The stopping criteria for the methods may vary. The procedure may employ a threshold on the measure used to determine when to stop the attribute selection process.

Q) What is the q-q plot and boxplot for given data.

A Q-Q (quantile-quantile) plot is a graphical tool used to compare two probability distributions by plotting their quantiles against each other. The main purpose of a Q-Q plot is to visually assess whether a set of data follows a particular distribution. In a Q-Q plot, if the points approximately fall along a straight line, it suggests that the data comes from the distribution being compared against.

A box plot, also known as a box-and-whisker plot, is a graphical summary of a dataset through five summary statistics: minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum. It provides a visual representation of the central tendency, spread, and skewness of the data. The box in the middle represents the interquartile range (IQR) between the first and third quartiles, with the median marked by a line inside the box. The "whiskers" extend to the minimum and maximum values within a certain range, typically 1.5 times the IQR from the first and third quartiles. Any data points beyond this range are considered outliers and are plotted individually.

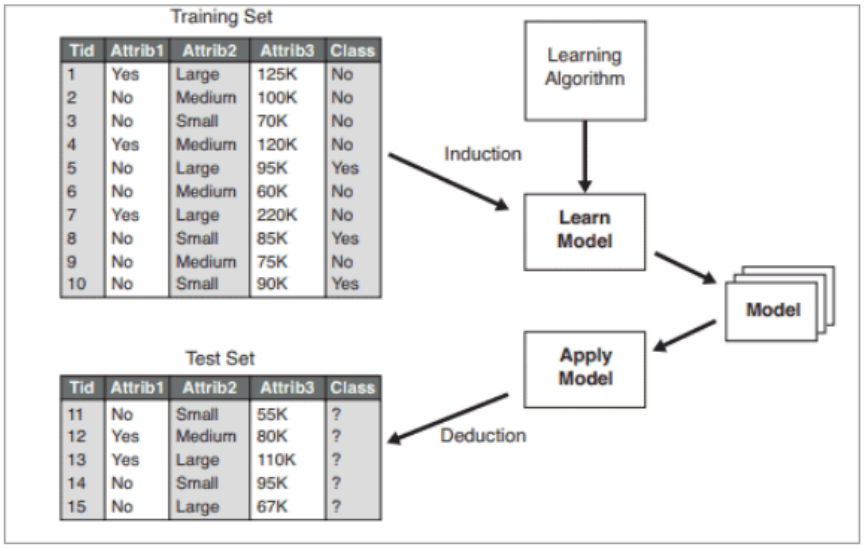
Module 3:

Classification:
Supervised and unsupervised learning
What is classification? classification applications
Classification
A process of finding a model that describes and distinguishes the data classes.

Predicts categorical class labels (discrete or nominal)
classification model building phases
A two-step process is followed, to build a classification model.

In the first step i.e. learning: A classification model based on training data is built.
In the second step i.e. Classification, the accuracy of the model is checked and then the model is used to classify new data. The class labels presented here are in the form of discrete values such as “yes” or “no”, “safe” or “risky”.

The general approach for building classification models is given below:

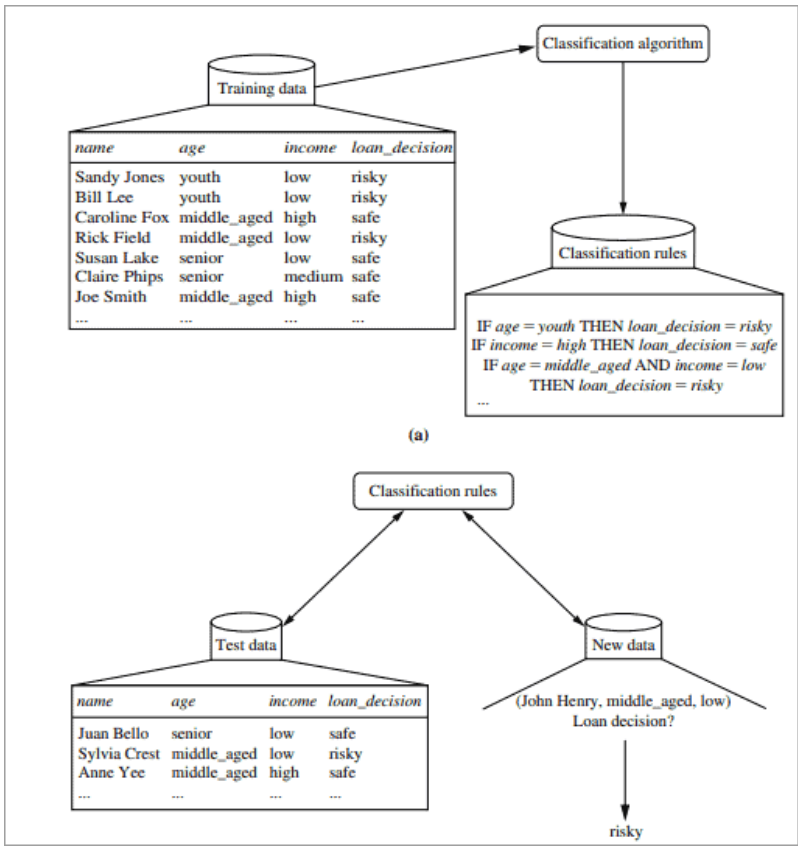


Classification algorithms
Explain the Decision tree-building process with an example.
Example of Creating a Decision Tree

(Example is taken from Data Mining Concepts: Han and Kimber)

#1) Learning Step: The training data is fed into the system to be analyzed by a classification algorithm. In this example, the class label is the attribute i.e. “loan decision”. The model built from this training data is represented in the form of decision rules.

#2) Classification: Test dataset are fed to the model to check the accuracy of the classification rule. If the model gives acceptable results then it is applied to a new dataset with unknown class variables.



Decision Tree algorithm

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, D.

Input:

- Data partition, D, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)      return N as a leaf node labeled with the class C;
(4)  if attribute_list is empty then
(5)      return N as a leaf node labeled with the majority class in D; // majority voting
(6)  apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
         multiway splits allowed then // not restricted to binary trees
(9)      attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)     let Dj be the set of data tuples in D satisfying outcome j; // a partition
(12)     if Dj is empty then
(13)         attach a leaf labeled with the majority class in D to node N;
(14)     else attach the node returned by Generate_decision_tree(Dj, attribute_list) to node N;
     endfor
(15) return N;
```

Entropy,Information Gain,Gain Ratio and Gini Index

Feature selection measures in building Decision Tree/splitting attribute selection Measure.

- An attribute selection measure is a heuristic for selecting the splitting criterion that "best" separates a given data partition, D , of class-labeled training tuples into individual classes.
- The attribute selection measure provides a ranking for each attribute describing the given training tuples.
- The three popular attribute selection measures—
- information gain
- gain ratio, and
- Gini index

Information Gain (ID3 algorithm)

- This measure is based on pioneering work by Claude Shannon on information theory, which studied the value or "information content" of messages.
- The attribute with the highest information gain is chosen as the splitting attribute for node N.
- This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or "impurity" in these partitions.
- Such an approach minimizes the expected number of tests needed to classify a given tuple and guarantees that a simple (but not necessarily the simplest) tree is found.
- Let pi be the probability that an arbitrary tuple in D belongs to class Ci , estimated by

  pi =|Ci, D|/|D|

- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)$$

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

- Select the attribute with the highest information gain.

Gain Ratio for Attribute Selection (C4.5)

- 'Information gain' measure is biased towards attributes with a large number of values(e.g. Unique ID attribute like Product_ID)

- C4.5 uses an extension to information gain known as gain ratio, which attempts to overcome this bias.
- It applies a kind of normalization to information gain using a "split information" value defined analogously with Info(D) as

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- This value represents the potential information generated by splitting the training data set, D, into v partitions, corresponding to the v outcomes of a test on attribute A.

- The gain ratio is defined as
- GainRatio(A) = Gain(A)/SplitInfoA(D)
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini index (CART)

Gini index (CART)

- Gini index measures the impurity of *D*, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

where p_i is the probability that a tuple in D belongs to class C_i and is estimated by $|C_{i,D}|/|D|$. The sum is computed over m classes.

- The Gini index considers a binary split for each attribute.

- **Case 1: A is discrete-valued:** To determine the best binary split on discrete valued attribute *A*, we examine all the possible subsets of the known values of *A*.
- Each subset, S_A, can be considered as a binary test for attribute *A* of the form "$A \in S_A$?"
- If *A* has *v* possible values, then there are 2^v possible subsets.
- Out of these, there are 2^v -2 possible ways to form two partitions of the data, *D*, based on a binary split on *A*.

31

Gini index (CART)

- When considering a binary split, we compute a weighted sum of the impurity of each resulting partition. For example, if a binary split on A partitions D into D_1 and D_2, the Gini index of D given that partitioning is

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2).$$

- For each attribute, each of the possible binary splits is considered.
- For a discrete-valued attribute, the subset that gives the **minimum Gini index** for that attribute is selected as its splitting subset.

- The reduction in impurity that would be incurred by a binary split on a discrete- or continuous-valued attribute A is

$$\Delta Gini(A) = Gini(D) - Gini_A(D).$$

- The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute. This attribute and its splitting subset (for a discrete-valued splitting attribute) together form the splitting criterion.

Gini index (Continuous-valued attribute)

- For continuous-valued attributes, each possible split-point must be considered.
- The strategy is to take the midpoint between each pair of (sorted) adjacent values as a possible split-point.
- The point giving the minimum Gini index for a given (continuous-valued) attribute is taken as the split-point of that attribute.

- Recall that for a possible split-point of *A*, *D*1 is the set of tuples in *D* satisfying *A* <= *split point*, and *D*2 is the set of tuples in *D* satisfying *A* > *split point*

- The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute. This attribute and its split-point (for a continuous-valued splitting attribute) together form the splitting criterion.

Different Metrics used for Evaluating Classifier Performance
Confusion matrix:

Metrics for Evaluating Classifier Performance

- Terminologies:
  - **Positive Tuples:** tuples of the main class of interest
  - **Negative Tuples:** All other tuples.
    (e.g. Given two classes , the positive tuples may be buys computer = yes while the negative tuples are buys computer = no.)
  - **True Positives(TP):**These refer to the positive tuples that were correctly labeled by the classifier.
  - **True Negatives(TN):** These are the negative tuples that were correctly labeled by the classifier.
  - **False Positives(FP):**These are the negative tuples that were incorrectly labeled as positive
    (e.g., tuples of class buys computer = no for which the classifier predicted buys computer = yes).
  - **False Negatives(FN):**These are the positive tuples that were mislabeled as negative (e.g., tuples of class buys computer = yes for which the classifier predicted buys computer = no).

These terms are summarized in a **Confusion Matrix**

The Classifier Evaluation Measures

Measure	*Formula*
accuracy, recognition rate	$\frac{TP+TN}{P+N}$
error rate, misclassification rate	$\frac{FP+FN}{P+N}$
sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
precision	$\frac{TP}{TP+FP}$
F, F_1, F-score, harmonic mean of precision and recall	$\frac{2 \times precision \times recall}{precision + recall}$
F_β, where β is a non-negative real number	$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

Metrics for Evaluating Classifier Performance

- **Confusion Matrix (CM):** The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes.

		Predicted class: *yes*	Predicted class: *no*	
Actual class	*yes*	*TP*	*FN*	*P*
	no	*FP*	*TN*	*N*
	Total	P'	N'	$P+N$

Given m classes, $CM_{i,j}$, an entry in a **confusion matrix**, indicates # of tuples in class i that are labeled by the classifier as class j

$$accuracy = \frac{TP + TN}{P + N}.$$

$$error\ rate = \frac{FP + FN}{P + N}.$$

3

Metrics for Evaluating Classifier Performance

		Predicted class: *yes*	Predicted class: *no*	
Actual class	*yes*	*TP*	*FN*	*P*
	no	*FP*	*TN*	*N*
	Total	P'	N'	$P+N$

classes	buy_computer = yes	buy_computer = no	total	recognition(%)
buy_computer = yes	6954	46	7000	99.34
buy_computer = no	412	2588	3000	86.27
total	7366	2634	10000	95.52

- Accuracy of a classifier M, acc(M): percentage of test set tuples that are correctly classified by the model $accuracy = \frac{TP + TN}{P + N}.$
- Error rate (misclassification rate) of M = 1 – acc(M) $error\ rate = \frac{FP + FN}{P + N}.$

4

Different Metrics Used for Evaluating imbalanced Classifier.

The main class of interest is rare.

e.g In medical data, there may be a rare class, such as “cancer.”

For Class Imbalance problems , we need other measures, which assesses how well the classifier can recognize the positive tuples (e.g. cancer = yes) and how well it can recognize the negative tuples (e.g. cancer = no).

Alternative Accuracy Measures: Sensitivity and Specificity

- **Sensitivity = True Positive Rate(TPR) = TP/P**
  /* true positive recognition rate:(i.e., the proportion of positive tuples that are correctly identified) */
- **Specificity = True Negative Rate(TNR) = TN/N**
  /* true negative recognition rate .(the proportion of negative tuples that are correctly identified*/

$$accuracy = sensitivity\frac{P}{(P+N)} + specificity\frac{N}{(P+N)}.$$

Classes	yes	no	Total	Recognition (%)
yes	90	210	300	30.00
no	140	9560	9700	98.56
Total	230	9770	10,000	96.40

- The sensitivity of the classifier is 90 / 300 = 30.00%.
- The specificity is 9560 / 9700 = 98.56%.
- The classifier's overall accuracy is 9650 / 10,000 = 96.50%.

Confusion matrix for the classes *cancer = yes* and *cancer = no*.

Metrics for Evaluating Classifier Performance
Precision and Recall

- **Precision** is a measure of *exactness* (i.e., what percentage of tuples labeled / predicted as positive are actually such),
- **Recall** is a measure of *completeness* (what percentage of positive tuples are labeled/predicted as such). It is similar to **sensitivity.**

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN} = \frac{TP}{P}.$$

- **Example:**

Classes	yes	no	Total	Recognition (%)
yes	90	210	300	30.00
no	140	9560	9700	98.56
Total	230	9770	10,000	96.40

- The precision of the classifier in Example for the *yes* class is 90 / 230 = 39.13%.
- The recall is 90 /300 = 30.00%, which is the same calculation for sensitivity
- There tends to be an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other.

Confusion matrix for the classes *cancer = yes* and *cancer = no*.

F1- Score

- F1-Measure provides a way to combine both precision and recall into a single measure that captures both properties.
- We can have excellent precision with terrible recall, or alternately, terrible precision with excellent recall.
- F1-measure provides a way to express both concerns with a single score which **weights precision and recall equally**.
- Once precision and recall have been calculated for a binary or multiclass classification problem, the two scores can be combined into the calculation of the F-Measure.

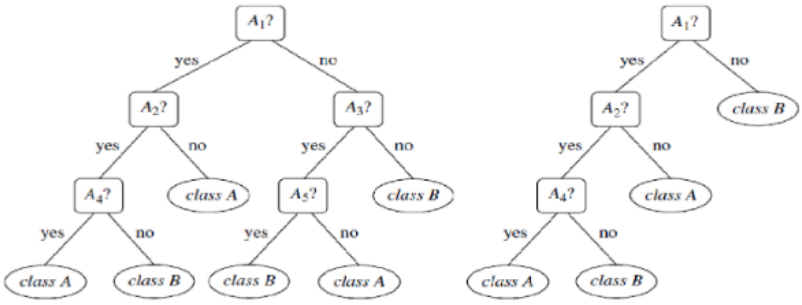
  F1-Score = (2 * Precision * Recall) / (Precision + Recall)

- The F1-score is the variant most often used when learning from imbalanced data.

Decision Tree Pruning

Overfitting and Tree Pruning

- Overfitting:
  - When a decision tree is built, many of the branches will reflect anomalies in the training data due to noise or outliers.
  - Tree pruning methods address this problem of *overfitting* the data.
  - Such methods typically use statistical measures to remove the least-reliable branches.



An unpruned decision tree and a pruned version of it.

Overfitting and Tree Pruning

- Two approaches to avoid overfitting : **Prepruning and Postpruning**
  - **Prepruning:**
    - Halt tree construction early (e.g., by deciding not to further split or partition the subset of training tuples at a given node).
    - Upon halting, the node becomes a leaf.
    - The leaf may hold the most frequent class among the subset tuples or the probability distribution of those tuples.
    - Difficult to choose an appropriate threshold
  - **Postpruning:**
    - Remove branches from a "fully grown" tree—A subtree at a given node is pruned by removing its branches and replacing it with a leaf.
    - The leaf is labeled with the most frequent class among the subtree being replaced.

41

Probd in DT and Rules generation from Decision tree
Naive Bayes Allems on Decision Tree:
Steps involvegorithm
Problems on Naive Bayes algorithm
Where do we use linear regression? Explain linear regression.

What is Linear Regression?

Linear regression is a method used to model the relationship between a dependent variable (usually denoted as y) and one or more independent variables (usually denoted as x). The relationship is modeled as a linear equation:

$$y = mx + b + \varepsilon$$

Where:

- y is the dependent variable (the one we want to predict).
- x is the independent variable (the one used to make predictions).
- m is the slope of the line (how much y changes for a unit change in x).
- b is the intercept (the value of y when x is 0).
- ε represents the error term, which accounts for the variability in y that cannot be explained by the linear relationship with x.

1. Prediction: Linear regression is often used for prediction tasks. Given a set of independent variables and their corresponding dependent variable values, the model can predict the dependent variable for new, unseen data.
2. Understanding Relationships: Linear regression helps in understanding the relationship between the independent and dependent variables. It quantifies how much the dependent variable changes when the independent variable changes.
3. Hypothesis Testing: Linear regression can be used to test hypotheses about the relationships between variables. For example, you might want to test if there is a significant relationship between advertising expenditure and sales.

4. Forecasting: Linear regression can also be used for forecasting future values of the dependent variable based on trends observed in historical data.

Applications:

- Economics: Predicting how changes in factors like interest rates or inflation affect GDP.
- Finance: Predicting stock prices based on various financial indicators.
- Medicine: Predicting patient outcomes based on factors like age, weight, and blood pressure.
- Marketing: Analyzing the impact of advertising spending on sales.
- Engineering: Predicting the strength of materials based on various physical properties.

Differentiate classification and Regression

Classification	Regression
In this problem statement, the target variables are discrete.	In this problem statement, the target variables are continuous.
Problems like Spam Email Classification, Disease prediction like problems are solved using Classification Algorithms.	Problems like House Price Prediction, Rainfall Prediction like problems are solved using regression Algorithms.
In this algorithm, we try to find the best possible decision boundary which can separate the two classes with the maximum possible separation.	In this algorithm, we try to find the best-fit line which can represent the overall trend in the data.
Evaluation metrics like Precision, Recall, and F1-Score are used here to evaluate the performance of the classification algorithms.	Evaluation metrics like Mean Squared Error, R2-Score, and MAPE are used here to evaluate the performance of the regression algorithms.
Here we face the problems like binary Classification or Multi-Class Classification problems.	Here we face the problems like Linear Regression models as well as non-linear models.
Input Data are Independent variables and categorical dependent variable.	Input Data are Independent variables and continuous dependent variable.
The classification algorithm's task mapping the input value of x with the discrete output variable of y.	The regression algorithm's task is mapping input value (x) with continuous output variable (y).
Output is Categorical labels.	Output is Continuous numerical values.
Objective is to Predict categorical/class labels.	Objective is to Predicting continuous numerical values.
Example use cases are Spam detection, image recognition, sentiment analysis	Example use cases are Stock price prediction, house price prediction, demand forecasting.
Examples of classification algorithms are: Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), K-Nearest Neighbors (K-NN), Naive Bayes, Neural Networks, K-Means Clustering, Multi-layer Perceptron (MLP), etc.	**Examples of regression algorithms are:** Linear Regression, Polynomial Regression, Ridge Regression, Lasso Regression, Support Vector Regression (SVR), Decision Trees for Regression, Random Forest Regression, K-Nearest Neighbors (K-NN) Regression, Neural Networks for Regression, etc.

State Bayes theorem. How can it be applied for data classification? b) With example explain Bayesian belief network.

Based on the following data determine the gender of a person having height 6 ft., weight 130 lbs. and foot size 8 in. (use Naive Bayes algorithm).

person	height (feet)	weight (lbs)	foot size (inches)
male	6.00	180	10
male	6.00	180	10
male	5.50	170	8
male	6.00	170	10
female	5.00	130	8
female	5.50	150	6
female	5.00	130	6
female	6.00	150	8

Q) What is Overfitting and Tree Pruning

Overfitting

- Overfitting is a common problem that needs to be handled while training a decision tree model.
- Overfitting occurs when a model fits too closely to the training data and may become less accurate when encountering new data or predicting future outcomes.
- In an overfit condition, a model memorizes the noise of the training data and fails to capture essential patterns.
- In decision trees, In order to fit the data (even noisy data), the model keeps generating new nodes and ultimately the tree becomes too complex to interpret. The decision tree predicts well for the training data but can be inaccurate for new data. If a decision tree model is allowed to train to its full potential, it can overfit the training data.
- Tree pruning methods address this problem of overfitting the data.

Tree Pruning

Pruning is a technique that removes parts of the decision tree and prevents it from growing to its full depth. Pruning removes those parts of the decision tree that do not have the power to classify instances. Pruning can be of two types — Pre-Pruning and Post-Pruning.

Two approaches to avoid overfitting : **Prepruning and Postpruning**

- **Prepruning:**
  - Halt tree construction early (e.g., by deciding not to further split or partition the subset of training tuples at a given node).
  - Upon halting, the node becomes a leaf.
  - The leaf may hold the most frequent class among the subset tuples or the probability distribution of those tuples.
    - Difficult to choose an appropriate threshold

- **Postpruning:**
  - Remove branches from a "fully grown" tree—A subtree at a given node is pruned by removing its branches and replacing it with a leaf.
    - The leaf is labeled with the most frequent class among the subtree being replaced.

Q) What is the Class Imbalance problem? Explain with Example.(GPT)
The Class Imbalance problem occurs when the distribution of classes in a dataset is highly skewed, meaning that one class is significantly more prevalent than the others. This imbalance can lead to challenges in training machine learning models, particularly when the minority class (the less frequent class) is of particular interest but may be overlooked due to its scarcity.

Here's an example to illustrate the Class Imbalance problem:
Imagine you're working on a project to develop a model for credit card fraud detection. In your dataset, you have information about thousands of credit card transactions, where the majority of transactions are legitimate (non-fraudulent), but only a tiny fraction are actually fraudulent.

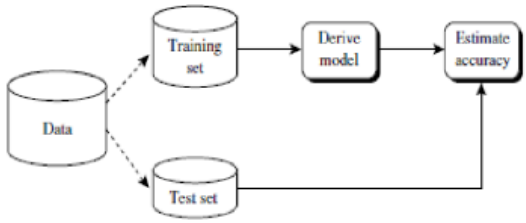
Let's say out of 10,000 transactions, only 50 are fraudulent. This means that the fraud class constitutes just 0.5% of the dataset, while the non-fraud class makes up the remaining 99.5%.

Now, if you were to train a machine learning model on this dataset without addressing the class imbalance, the model might simply learn to always predict the majority class (non-fraud) because it achieves high accuracy by doing so. As a result, the model may completely fail to identify instances of fraud, which is the critical task in this scenario.

Q) What are the different Methods for evaluating accuracy of the classifier(Holdout method,Random Subsampling,Cross Validation, Bootstrap method)
Holdout Method:

In the Holdout method, the dataset is divided into two parts: a training set and a testing set.
The classifier is trained on the training set and then evaluated on the separate testing set.
The evaluation metrics, such as accuracy, are calculated based on the performance of the classifier on the testing set.
Typically, a common split ratio is 70% training data and 30% testing data, but this can vary depending on the size and nature of the dataset.

- Holdout method
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation ---Pessimistic estimate



Estimating accuracy with the holdout method.

  - Random subsampling: a variation of holdout
    - Repeat holdout method k times, accuracy = avg. of the accuracies obtained from each iteration

Random Subsampling:
Similar to the Holdout method, but instead of performing a single split of the dataset into training and testing sets, Random Subsampling involves repeating this process multiple times.
Each time, a random subset of the data is selected as the training set, and the remaining data is used as the testing set.
The evaluation metrics are then averaged over all iterations to obtain a more stable estimate of the classifier's performance.

Cross-Validation:
Cross-Validation is a more robust method compared to Holdout and Random Subsampling.
It involves partitioning the dataset into k equal-sized folds (or subsets).
The classifier is trained on k-1 folds and tested on the remaining fold, iteratively for k times (each fold serves as the testing set once).
The evaluation metrics are averaged over all iterations to obtain a comprehensive assessment of the classifier's performance.

Common variants include k-fold cross-validation and stratified k-fold cross-validation, where class distributions are preserved in each fold.

Cross-validation (*k*-fold, where k = 10 is most popular)

- Randomly partition the initial data into *k mutually exclusive* subsets or folds,D1 to Dk, each of approximately equal size.
- Perform training and testing k times.
- At *i*-th iteration, use D_i as test set and others collectively as training set
- Unlike the holdout and random subsampling methods, here each sample is used the same number of times for training and once for testing.
- The results of each iteration are averaged, to find accuracy which is used as a performance metric to compare the efficiency of different models.
- The k-fold cross-validation technique generally produces less biased models as every data point from the original dataset will appear in both the training and testing set.
- This method is optimal if you have a limited amount of data.

Leave-one-out: A special case of k folds where k is set to number of initial tuples i.e only one sample is left out at a time for test set.

Stratified cross-validation: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data.

Bootstrap Method:

Bootstrap is a resampling technique that involves generating multiple datasets of the same size as the original dataset by randomly sampling with replacement.

Each bootstrap sample is used to train and test the classifier.

The evaluation metrics are then averaged over all bootstrap samples to obtain a robust estimate of the classifier's performance.

Bootstrap is particularly useful when the dataset is limited in size or when the distribution of the data is complex and not easily represented by a simple parametric model.

- **Bootstrap**
  - Works well with small data sets
  - Samples the given training tuples uniformly *with replacement*
    - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is **.632 bootstrap**
  - Suppose we are given a data set of d tuples. The data set is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. On an average 63.2% of the original data will end up in the bootstrap sample, and the remaining 36.8% will form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$)
  - Repeat the sampling procedue k times, overall accuracy of the model:

$$Acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \times Acc(M_i)_{test\_set} + 0.368 \times Acc(M_i)_{train\_set}),$$

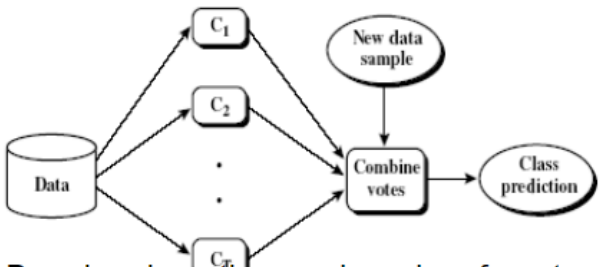
where $Acc(M_i)_{test\_set}$ is the accuracy of the model obtained with bootstrap sample i when it is applied to test set i. $Acc(M_i)_{train\_set}$ is the accuracy of the model obtained with bootstrap sample i when it is applied to the original set of data tuples.

4

Q) Explain the Ensemble Methods for Improving the Accuracy of classifier(Bagging,boosting, and random forest)

Ensemble methods are techniques that combine multiple base classifiers to improve the overall performance and accuracy of the model. Three popular ensemble methods are Bagging, Boosting, and Random Forest. Let's delve into each of them:

- Ensemble methods
  - Use a combination of models to increase accuracy
  - Combine a series of k learned models, M_1, M_2, …, M_k, with the aim of creating an improved model M*
  - A given data set, *D*, is used to create *k* training sets, *D*1, *D*2, : : : , *Dk*, where *Di(1<=i<=k-1)* is used to generate classifier *Mi* .
  - Given a new data tuple to classify, the base classifiers each vote by returning a class prediction. The ensemble returns a class prediction based on the votes of the base classifiers.
  - Ensembles yield better results when there is significant diversity among the models.



  - Examples are Bagging, boosting, and random forest

5

Bagging (Bootstrap Aggregating):

Bagging involves creating multiple subsets of the original dataset by sampling with replacement (bootstrap sampling).
Each subset is used to train a base classifier independently.
Predictions from all base classifiers are then aggregated, typically by averaging (for regression) or voting (for classification).
Bagging helps to reduce variance and overfitting by creating diverse base classifiers, each trained on a slightly different subset of the data.
The most famous example of Bagging is the Random Forest algorithm.

Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors' majority vote
- Training
  - Given a set D of *d* tuples, at each iteration *i*, a training set D_i of *d* tuples is sampled with replacement from D (i.e., bootstrap)
  - A classifier model M_i is learned for each training set D_i
- To classify an unknown sample **X** ,
  - Each classifier M_i returns its class prediction which counts as one vote.
  - The bagged classifier M*, counts the votes and assigns the class with the most votes to **X**
- Bagging can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
  - Often significant accuracy than a single classifier derived from D
  - It will not be considerably worse and is more robust to the effects of noisy data and overfitting.
  - The increased accuracy occurs because the composite model reduces the variance of the individual classifiers

6

Bagging: Bootstrap Aggregation

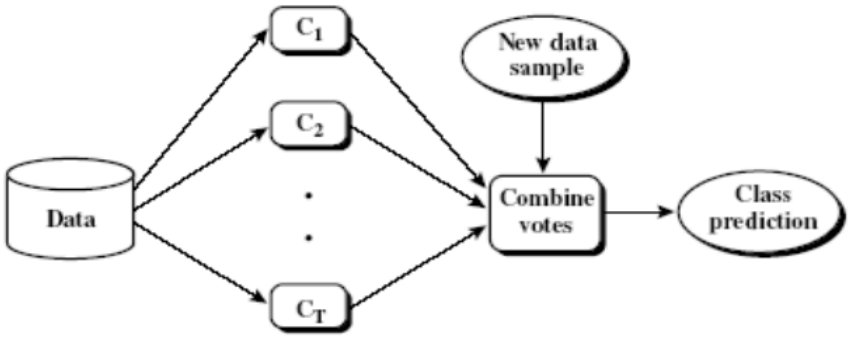


Figure: Increasing classifier accuracy: Ensemble methods generate a set of classification models, M1, M2, : : : , Mk. Given a new data tuple to classify, each classifier "votes" for the class label of that tuple. The ensemble combines the votes to return a class prediction.

Algorithm: Bagging. The bagging algorithm—create an ensemble of classification models for a learning scheme where each model gives an equally weighted prediction.

Input:

- D, a set of d training tuples;
- k, the number of models in the ensemble;
- a classification learning scheme (decision tree algorithm, naïve Bayesian, etc.).

Output: The ensemble—a composite model, $M*$.

Method:

(1) **for** $i = 1$ to k **do** // create k models:
(2) create bootstrap sample, D_i, by sampling D with replacement;
(3) use D_i and the learning scheme to derive a model, M_i;
(4) **endfor**

To use the ensemble to classify a tuple, X:

let each of the k models classify X and return the majority vote;

Boosting:

Boosting is an iterative ensemble method where base classifiers are trained sequentially, and each subsequent classifier focuses on the instances that were misclassified by the previous ones.

At each iteration, the weights of misclassified instances are adjusted to emphasize the importance of those instances in subsequent iterations.

Boosting algorithms aim to improve model performance by giving more weight to difficult-to-classify instances, effectively reducing bias and increasing overall accuracy.

Popular boosting algorithms include AdaBoost (Adaptive Boosting), Gradient Boosting Machines (GBM), and XGBoost.

Boosting

- How boosting works?
  - Weights are assigned to each training tuple
  - A series of k classifiers is iteratively learned
  - After a classifier M_i is learned, the weights are updated to allow the subsequent classifier, M_{i+1}, to pay more attention to the training tuples that were misclassified by M_i
  - The final boosted classifier M* combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy

Random Forest:

Random Forest is an ensemble learning method that combines the concepts of Bagging and decision trees.

It creates an ensemble of decision trees where each tree is trained on a random subset of the features and a random subset of the data (using bootstrap sampling).
During tree construction, at each node, the best split is chosen from a random subset of features, leading to greater diversity among individual trees.
The final prediction is made by averaging (for regression) or voting (for classification) over all trees in the forest.
Random Forest is highly effective due to its ability to handle high-dimensional data, reduce overfitting, and provide robust predictions.

Q) What is Simple Linear Regression and multiple linear regression? Examples .

Parameter	Linear (Simple) Regression	Multiple Regression
Definition	Models the relationship between one dependent and one independent variable.	Models the relationship between one dependent and two or more independent variables.
Equation	$Y = C_0 + C_1X + e$	$Y = C_0 + C_1X_1 + C_2X_2 + C_3X_3 + ... + C_nX_n + e$
Complexity	Simpler dealing with one relationship.	More complex due to multiple relationships.
Use Cases	Suitable when there is one clear predictor.	Suitable when multiple factors affect the outcome.
Assumptions	Linearity, Independence, Homoscedasticity, Normality	Same as linear regression, with the added concern of multicollinearity.
Visualization	Typically visualized with a 2D scatter plot and a line of best fit.	Requires 3D or multi-dimensional space, often represented using partial regression plots.
Risk of Overfitting	Lower, as it deals with only one predictor.	Higher, especially if too many predictors are used without adequate data.
Multicollinearity Concern	Not applicable, as there's only one predictor.	A primary concern; having correlated predictors can affect the model's accuracy and interpretation.
Applications	Basic research, simple predictions, understanding a singular relationship.	Complex research, multifactorial predictions, studying interrelated systems.

Module 4

Clustering:

Q) Clustering process

Clustering or cluster analysis is a machine learning technique, which groups the unlabelled dataset. It can be defined as "A way of grouping the data points into different clusters, consisting of similar data points. The objects with the possible similarities remain in a group that has less or no similarities with another group."
It does it by finding some similar patterns in the unlabelled dataset such as shape, size, color, behavior, etc., and divides them as per the presence and absence of those similar patterns.
It is an unsupervised learning method, hence no supervision is provided to the algorithm, and it deals with the unlabeled dataset.

The clustering technique can be widely used in various tasks. Some most common uses of this technique are:

1. Market Segmentation

2. Statistical data analysis
3. Social network analysis
4. Image segmentation
5. Anomaly detection, etc.

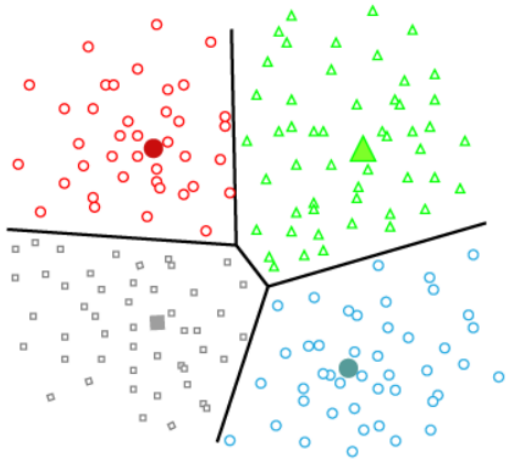
Types of Clustering Methods

The clustering methods are broadly divided into Hard clustering (datapoint belongs to only one group) and Soft Clustering (data points can belong to another group also). But there are also other various approaches of Clustering exist. Below are the main clustering methods used in Machine learning:

1. Partitioning Clustering

It is a type of clustering that divides the data into non-hierarchical groups. It is also known as the centroid-based method. The most common example of partitioning clustering is the K-Means Clustering algorithm.

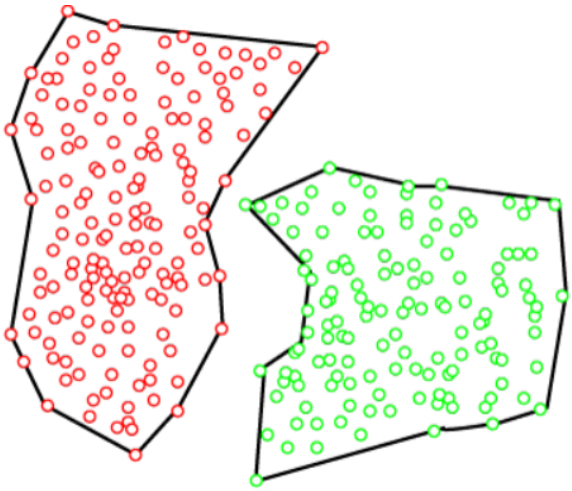
In this type, the dataset is divided into a set of k groups, where K is used to define the number of pre-defined groups. The cluster center is created in such a way that the distance between the data points of one cluster is minimum as compared to another cluster centroid.



2. Density-Based Clustering

The density-based clustering method connects the highly-dense areas into clusters, and the arbitrarily shaped distributions are formed as long as the dense region can be connected. This algorithm does it by identifying different clusters in the dataset and connects the areas of high densities into clusters. The dense areas in data space are divided from each other by sparser areas.

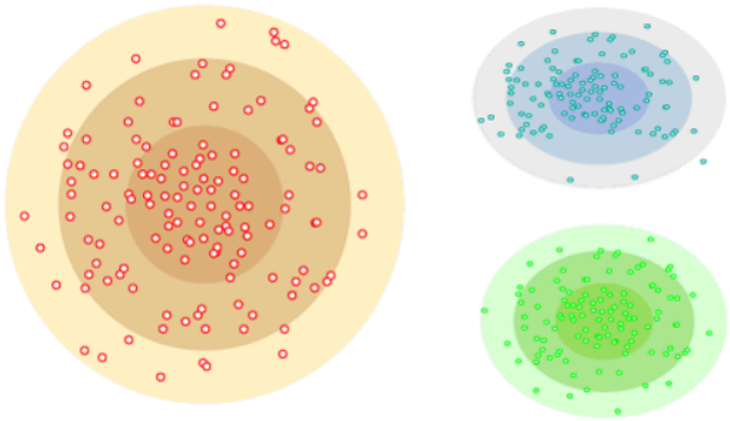
These algorithms can face difficulty in clustering the data points if the dataset has varying densities and high dimensions.



3. Distribution Model-Based Clustering

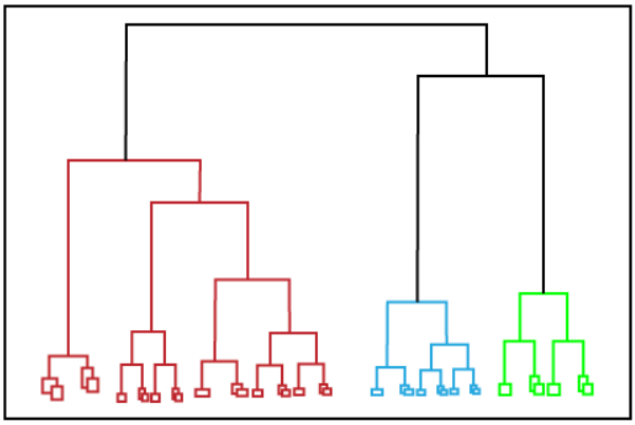
In the distribution model-based clustering method, the data is divided based on the probability of how a dataset belongs to a particular distribution. The grouping is done by assuming some distributions commonly Gaussian Distribution.

The example of this type is the Expectation-Maximization Clustering algorithm that uses Gaussian Mixture Models (GMM).



4. Hierarchical Clustering

Hierarchical clustering can be used as an alternative for the partitioned clustering as there is no requirement of pre-specifying the number of clusters to be created. In this technique, the dataset is divided into clusters to create a tree-like structure, which is also called a dendrogram. The observations or any number of clusters can be selected by cutting the tree at the correct level. The most common example of this method is the Agglomerative Hierarchical algorithm.



5. Fuzzy Clustering

Fuzzy Clustering

Fuzzy clustering is a type of soft method in which a data object may belong to more than one group or cluster. Each dataset has a set of membership coefficients, which depend on the degree of membership to be in a cluster. Fuzzy C-means algorithm is the example of this type of clustering; it is sometimes also known as the Fuzzy k-means algorithm.

Clustering Algorithms

1. K-Means algorithm: The k-means algorithm is one of the most popular clustering algorithms. It classifies the dataset by dividing the samples into different clusters of equal variances. The number of clusters must be specified in this algorithm. It is fast with fewer computations required, with the linear complexity of O(n).
2. Mean-shift algorithm: Mean-shift algorithm tries to find the dense areas in the smooth density of data points. It is an example of a centroid-based model, that works on updating the candidates for centroid to be the center of the points within a given region.
3. DBSCAN Algorithm: It stands for Density-Based Spatial Clustering of Applications with Noise. It is an example of a density-based model similar to the mean-shift, but with some remarkable advantages. In this algorithm, the areas of high density are separated by the areas of low density. Because of this, the clusters can be found in any arbitrary shape.
4. Expectation-Maximization Clustering using GMM: This algorithm can be used as an alternative for the k-means algorithm or for those cases where K-means can be failed. In GMM, it is assumed that the data points are Gaussian distributed.
5. Agglomerative Hierarchical algorithm: The Agglomerative hierarchical algorithm performs the bottom-up hierarchical clustering. In this, each data point is treated as a single cluster at the outset and then successively merged. The cluster hierarchy can be represented as a tree-structure. (Divisive bhi)
6. Affinity Propagation: It is different from other clustering algorithms as it does not require to specify the number of clusters. In this, each data point sends a message between the pair of data points until convergence. It has O(N2T) time complexity, which is the main drawback of this algorithm.

Q) Explain different types of clustering techniques

Upar wale Q me cover hogaya

Q) K-means algorithm and problems based on K-means.

The *K-Means* Clustering Method

- A centroid-based partitioning technique uses the ***centroid*** of a cluster, *Ci* , to represent that cluster.
- The quality of cluster *Ci* can be measured by the **within cluster variation**, which is the sum of *squared error* between all objects in *Ci* and the centroid ***ci***, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(\boldsymbol{p}, \boldsymbol{c_i})^2,$$

where *E* is the sum of the squared error for all objects in the data set; ***p*** is the point in space representing a given object; and ***ci*** is the centroid of cluster *Ci* (both ***p*** and ***ci*** are multidimensional).

- This objective function tries to make the resulting k clusters as compact and as separate as possible.
- Optimizing the within-cluster variation is computationally challenging.
- To overcome the prohibitive computational cost for the exact solution, greedy approaches are often used in practice. e.g k-means algorithm

*

8

- Given *k*, the *k-means* algorithm is implemented as below :
  - First, it randomly selects *k* of the objects in *D*, each of which initially represents a cluster mean or cluster center.
  - For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the Euclidean distance between the object and the cluster mean.
  - The *k*-means algorithm then iteratively improves the within-cluster variation.
    - For each cluster, it computes the new mean using the objects assigned to the cluster in the previous iteration.
    - All the objects are then reassigned using the updated means as the new cluster centers.
    - The iterations continue until the assignment is stable, that is, the clusters formed in the current round are the same as those formed in the previous round.

The *K-Means* Clustering Method **(5 steps)**

Step 1: Choose the no.of clusters ,k.
Step2: Select at random k points, the centroids(not necessarily from your dataset)

Step3: Assign each data point to the closest centroid as that forms k clusters.

Step4: Compute and place new centroid of each cluster.

Step5: Reassign each data point to the new closest centroid . If any reassignment takes place go to step 4 . Otherwise (if the assignment is stable, i.e. the clusters formed in the current round are the same as those formed in the previous round) then STOP.

- The process of iteratively re assigning objects to clusters to improve the partitioning is referred to as ***iterative relocation technique.***

Problems with K-means

1. Sensitivity to Initial Centroids: The final clustering results can be significantly influenced by the initial positions of the centroids. Running K-means multiple times with different initial values can help mitigate this issue, but it requires careful consideration of the initial centroids.
2. Outliers: K-means can be sensitive to outliers, which can distort the clustering process and lead to less reliable clusters. Outliers can either drag the centroids away from the main cluster or create their own clusters, affecting the overall clustering outcome.
3. Assumption of Round Clusters: K-means assumes that clusters are spherical and have roughly the same size. This assumption may not hold true for real-world

data, where clusters can have different shapes and sizes. This limitation can result in less accurate clustering.

4. Determining the Number of Clusters: The algorithm requires the user to specify the number of clusters (k) in advance. Choosing an incorrect number of clusters can lead to misleading results. Techniques like the elbow method or silhouette analysis can help estimate the appropriate number of clusters, but it remains a challenge.
5. Computational Complexity: K-means can become computationally expensive and slow as the number of data points increases. For very large datasets, alternative techniques like Mini-Batch K-means or distributed frameworks may be necessary to handle the scaling issue.
6. Dimensionality Issues: As the number of dimensions increases, the distance-based similarity measure used by K-means can converge to a constant value between any given examples, making it less effective. Dimensionality reduction techniques, such as Principal Component Analysis (PCA), can be used to address this issue.
7. Global Maximum Issue: Finding the global minimum for the sum of squared distances (S.O.D) is fundamentally NP-Hard, making it computationally expensive to find the true minimum. The Lloyd's algorithm, which K-means is based on, is an approximation that can lead to suboptimal clustering results.
8. Ease of Misuse: K-means is easy to use but can lead to misleading results if not properly understood or applied. It's important to have a good understanding of the data and the assumptions of the algorithm to avoid misinterpretations.

Q) What are the weaknesses of hierarchical clustering?

Hierarchical clustering, while a powerful tool for data analysis, has several weaknesses that can limit its effectiveness in certain applications. Here's a summary of the main weaknesses based on the provided sources:

1. Sensitivity to Noise and Outliers: Hierarchical clustering can be significantly affected by noise and outliers in the data. These can distort the clustering process and lead to less reliable clusters.
2. Difficulty Handling Different Sized Clusters and Convex Shapes: The algorithm may struggle with data that contains clusters of varying sizes or clusters that are not spherical. This can result in less accurate clustering outcomes.
3. Non-Iterative and Greedy Nature:
4. Hierarchical clustering is a non-iterative, single-pass greedy algorithm. This means that it optimizes the current step's task without necessarily guaranteeing the best partition at a distant future step. This characteristic can lead to suboptimal clustering results.
5. Inability to Undo Decisions: Once a decision is made to combine two clusters, it cannot be undone. This rigidity can limit the flexibility of the clustering process and make it difficult to adjust the clustering structure after it has been initially established.

6. Performance on Large Datasets: Hierarchical clustering does not perform well on very large datasets. The computational complexity and memory requirements can make it impractical for analyzing large volumes of data.
7. Dependence on the Order of Data: The order in which data points are processed can significantly impact the final clustering results. This dependence on the initial data order can lead to different clustering outcomes depending on the starting point.
8. Issues with Missing Data: Most hierarchical clustering software does not work well with datasets that contain missing values. This limitation can restrict the applicability of hierarchical clustering in real-world scenarios where missing data is common.
9. Misinterpretation of Dendrograms: The main output of hierarchical clustering, the dendrogram, is often misinterpreted. This can lead to incorrect conclusions about the data structure and relationships between data points.

Q) Compare k-means with k-medoids algorithms for clustering.

K-means	K-medoids
K-means takes the mean of data points to create new points called centroids.	K-medoids uses points from the data to serve as points called medoids.
Centroids are new points previously not found in the data.	Medoids are existing points from the data.
K-means can only by used for numerical data.	K-medoids can be used for both numerical and categorical data.
K-means focuses on reducing the sum of squared distances, also known as the sum of squared error (SSE).	K-medoids focuses on reducing the dissimilarities between clusters of data from the dataset.
K-means uses Euclidean distance.	K-medoids uses Manhattan distance.
K-means is not sensitive to outliers within the data.	K-medoids is outlier resistant and can reduce the effect of outliers.
K-means does not cater to noise in the data.	K-medoids effectively reduces the noise in the data.
K-means is less costly to implement.	K-medoids is more costly to implement.
K-means is faster.	K-medoids is comparatively not as fast.

Q) What is the main objective of clustering? Give the categorization of clustering approaches. Briefly discuss them.

The main objective of clustering in machine learning is to group similar data points together into clusters, thereby discovering meaningful structure, explaining underlying processes, and identifying descriptive attributes and groupings within a dataset. Clustering is an unsupervised learning problem, meaning it does not require labeled data for training. Instead, it identifies patterns and trends within the data, allowing for the creation of discrete classes or clusters based on the similarity of the data points.

Categorization of clustering approaches:

1. Partitioning Methods:
   Partitioning methods divide the data into non-overlapping clusters, where each data point belongs to exactly one cluster. Examples include K-means and K-medoids (PAM).
2. Hierarchical Methods:
   Hierarchical methods create a tree-like structure of clusters, where clusters at higher levels of the hierarchy contain clusters from lower levels. Agglomerative hierarchical clustering merges similar clusters iteratively, while divisive hierarchical clustering splits clusters recursively.
3. Density-Based Methods:
   Density-based methods group data points based on their density in the data space. Clusters are formed around areas of high data density, separated by regions of lower density. DBSCAN and OPTICS are examples of density-based clustering algorithms.
4. Distribution-Based Methods:
   Distribution-based methods assume that the data is generated from a mixture of probability distributions. These methods model clusters as distributions and use statistical techniques to estimate the parameters of these distributions. Gaussian Mixture Models (GMM) and Expectation-Maximization (EM) clustering are common examples.
5. Centroid-Based Methods:
   Centroid-based methods represent each cluster by a central point (centroid) and assign data points to the nearest centroid. These methods are based on the notion of similarity between data points and centroids. K-means, K-medians, and Fuzzy C-means (FCM) are examples of centroid-based clustering algorithms.

Q) Differentiate between AGNES and DIANA algorithms. b) How to access the cluster quality?

Parameters	Agglomerative Clustering	Divisive Clustering
Category	Bottom-up approach	Top-down approach
Approach	each data point starts in its own cluster, and the algorithm recursively merges the closest pairs of clusters until a single cluster containing all the data points is obtained.	all data points start in a single cluster, and the algorithm recursively splits the cluster into smaller sub-clusters until each data point is in its own cluster.
Complexity level	Agglomerative clustering is generally more computationally expensive, especially for large datasets as this approach requires the calculation of all pairwise distances between data points, which can be computationally expensive.	Comparatively less expensive as divisive clustering only requires the calculation of distances between sub-clusters, which can reduce the computational burden.
Outliers	Agglomerative clustering can handle outliers better than divisive clustering since outliers can be absorbed into larger clusters	divisive clustering may create sub-clusters around outliers, leading to suboptimal clustering results.
Interpretability	Agglomerative clustering tends to produce more interpretable results since the dendrogram shows the merging process of the clusters, and the user can choose the number of clusters based on the desired level of granularity.	divisive clustering can be more difficult to interpret since the dendrogram shows the splitting process of the clusters, and the user must choose a stopping criterion to determine the number of clusters.
Implementation	Scikit-learn provides multiple linkage methods for agglomerative clustering, such as "ward," "complete," "average," and "single,"	divisive clustering is not currently implemented in Scikit-learn.
Example	Here are some of the applications in which Agglomerative Clustering is used : Image segmentation, Customer segmentation, Social network analysis, Document clustering, Genetics, genomics, etc., and many more.	Here are some of the applications in which Divisive Clustering is used : Market segmentation, Anomaly detection, Biological classification, Natural language processing, etc.

Accessing cluster quality involves evaluating the effectiveness of the clustering algorithm in producing meaningful and useful clusters. Several methods can be used to assess cluster quality:

1. Internal Evaluation Metrics: These metrics evaluate the compactness and separation of clusters based solely on the input data and the resulting clustering. Examples include the silhouette score, Davies–Bouldin index, and Dunn index.
2. External Evaluation Metrics: These metrics compare the clustering results to known ground truth labels, if available. Examples include purity, F-measure, and Rand index.
3. Visual Inspection: Visualizing the clustering results can provide insights into the structure of the data and the quality of the clusters. Techniques such as scatter plots, dendrograms, and heatmaps can be used for visual inspection.
4. Domain-specific Evaluation: In some cases, domain-specific knowledge or expertise may be used to evaluate the quality of clusters. This involves assessing whether the clusters obtained align with the expected patterns or characteristics of the data.

By assessing cluster quality using one or more of these methods, you can gain insights into the effectiveness of the clustering algorithm and make informed decisions about the number of clusters and the interpretation of the results.

Q) inter-cluster distance using single linkage,complete linkage and average linkage measure

Inter-cluster distance measures, such as single linkage, complete linkage, and average linkage, are used in hierarchical clustering algorithms to determine the distance between clusters. These measures play a crucial role in deciding which clusters to merge or split during the clustering process. Here's a brief explanation of each:

Single Linkage:

- Also known as the minimum linkage or nearest neighbor method.
- It calculates the distance between two clusters based on the shortest distance between any two points in the two clusters.
- The distance between two clusters is defined as the minimum of all pairwise distances between points in the two clusters.
- Single linkage tends to produce elongated clusters because it is sensitive to outliers and noise.
- It is computationally efficient but can lead to the "chaining" effect, where clusters are connected in a chain-like manner.

Complete Linkage:

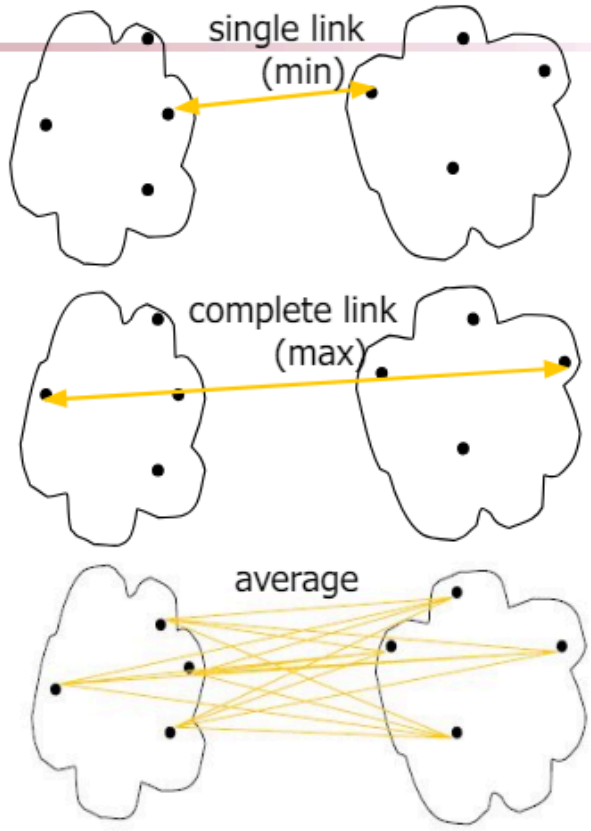
- Also known as maximum linkage or farthest neighbor method.
- It calculates the distance between two clusters based on the maximum distance between any two points in the two clusters.
- The distance between two clusters is defined as the maximum of all pairwise distances between points in the two clusters.
- Complete linkage tends to produce compact, spherical clusters because it focuses on the maximum distance between points.
- It is less sensitive to outliers compared to single linkage but can be computationally more expensive.

Average Linkage:

- Also known as mean linkage method.
- It calculates the distance between two clusters based on the average distance between all pairs of points in the two clusters.
- The distance between two clusters is defined as the average of all pairwise distances between points in the two clusters.
- Average linkage strikes a balance between single and complete linkage and tends to produce clusters of moderate compactness.
- It is less sensitive to outliers compared to single linkage and less prone to the chaining effect, making it a popular choice in practice.
- Average linkage can be computationally more expensive than single linkage but less so than complete linkage.

Cluster Distance Measures

- **Single link**: smallest distance between an element in one cluster and an element in the other, i.e., $d(C_i, C_j) = \min\{d(x_{ip}, x_{jq})\}$
- **Complete link**: largest distance between an element in one cluster and an element in the other, i.e., $d(C_i, C_j) = \max\{d(x_{ip}, x_{jq})\}$
- **Average**: avg distance between elements in one cluster and elements in the other, i.e., $d(C_i, C_j) = \text{avg}\{d(x_{ip}, x_{jq})\}$



Hierarchical clustering:

Q) Explain Agglomerative (AGNES) and Divisive (DIANA) algorithm) Difference me se hi padhlo :P

Q) Compare Agglomerative (AGNES) and Divisive (DIANA) algorithm)

Dendrogram and cluster formation from dendogram

A dendrogram is a tree-like diagram that illustrates the arrangement of clusters in hierarchical clustering. It shows how individual data points are grouped into clusters at different levels of similarity. Each node in the dendrogram represents a cluster, and the height of the node represents the dissimilarity or distance between clusters. Dendrograms are commonly used to visualize the hierarchical structure of clusters and aid in determining the appropriate number of clusters.

Here's how a dendrogram is typically formed and how clusters are identified from it:

1. Calculation of Pairwise Distances:
   First, pairwise distances between all data points are calculated based on a chosen distance metric, such as Euclidean distance or Manhattan distance.
2. Hierarchical Clustering Algorithm:

A hierarchical clustering algorithm, such as agglomerative hierarchical clustering (e.g., AGNES) or divisive hierarchical clustering (e.g., DIANA), is applied to the data using the pairwise distances.

3. In agglomerative clustering, the algorithm starts with each data point as a separate cluster and iteratively merges the closest clusters until all points belong to a single cluster. The dendrogram is built during this process, with each merge represented by a branch in the tree.
4. In divisive clustering, the algorithm starts with all data points in a single cluster and recursively divides clusters into smaller clusters until each cluster contains only one point. Again, the dendrogram is constructed as clusters are split.
5. Dendrogram Visualization:
   The dendrogram is visualized as a tree-like structure, with data points at the bottom and clusters formed at each level of the hierarchy.
6. The height of each node in the dendrogram represents the dissimilarity or distance between the clusters being merged or split. Longer branches indicate greater dissimilarity.
   Cluster Formation:
7. To identify clusters from the dendrogram, a horizontal line is drawn at a certain height, known as the "cut-off" or "threshold" level.
8. Clusters are formed by cutting the dendrogram at this threshold level. Each cluster is composed of all data points below the cut-off line, with branches representing the hierarchy of sub-clusters within each cluster.
9. Determining the Number of Clusters:
10. The appropriate number of clusters can be determined by selecting a cut-off level that balances the desire for a meaningful number of clusters with the need to avoid over-segmentation or under-segmentation.

Methods such as the elbow method, silhouette score, or inspecting the dendrogram visually can help in selecting an optimal number of clusters.

Q) What is the goal of clustering? How does partitioning around medoids algorithm achieve this goal?

The goal of clustering is to group similar data points together based on certain criteria or features, without prior knowledge of the group labels. Clustering aims to discover natural patterns, structures, or relationships within the data and to organize it into meaningful clusters.

The Partitioning Around Medoids (PAM) algorithm, also known as K-medoids, is a clustering algorithm that aims to achieve this goal by iteratively partitioning the data into K clusters, where K is specified by the user. Here's how the PAM algorithm achieves clustering:

1) Initialization:

The algorithm starts by randomly selecting K data points from the dataset as initial cluster representatives, called medoids. These initial medoids can be chosen randomly or using a heuristic method.

2) Assignment:

Each data point is assigned to the nearest medoid based on a chosen distance metric, such as Euclidean distance or Manhattan distance. The assignment is based on the similarity between the data point and the medoids.

3) Update:

For each cluster, the algorithm evaluates whether swapping a non-medoid data point with one of the medoids would improve the clustering quality. This is done by calculating the total dissimilarity (e.g., sum of distances) between the data points in the cluster and the medoid.

If swapping a non-medoid with a medoid reduces the total dissimilarity, the medoid is updated to the new data point. This step ensures that the medoids are representative of their respective clusters.

4) Iteration:

Steps 2 and 3 are repeated iteratively until convergence, i.e., until no further improvement in clustering quality can be achieved by updating the medoids.

5) Output:

The final clustering result consists of K clusters, with each cluster represented by its medoid.

The PAM algorithm differs from the more commonly known K-means algorithm in that it uses actual data points (medoids) as cluster representatives instead of the mean of the data points (centroids). This makes PAM more robust to outliers and noise in the data, as medoids are less sensitive to extreme values compared to centroids.

Q) DBSCAN clustering ,BIRCH

BIRCH

BIRCH (1996)

- Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) is designed for clustering a large amount of numeric data by integrating hierarchical clustering (at the initial *microclustering* stage) and other clustering methods such as iterative partitioning (at the later *macroclustering* stage).
- Small summary of large dataset.

- It overcomes the two difficulties in agglomerative clustering methods: (1) scalability and (2) the inability to undo what was done in the previous step.

- BIRCH uses the notions of *clustering feature* to summarize a cluster, and *clustering feature tree* (*CF-tree*) to represent a cluster hierarchy.

- These structures help the clustering method to achieve good speed and scalability in large DS and also make it effective for incremental and dynamic clustering of incoming data objects.

- A clustering feature is essentially a summary of the statistics for the given cluster.
- The **clustering feature** (**CF**) of the cluster is a 3-D vector summarizing information about clusters of objects.
  - *CF* =(*n*,*LS*,*SS*) *where*
    - *LS* is the linear sum of the *n* points i.e., $\sum_{i=1}^{n} \boldsymbol{x}_i$
    - *SS* is the square sum of the data points i.e., $\sum_{i=1}^{n} \boldsymbol{x}_i^2$
- Using a clustering feature, we can easily derive many useful statistics of a cluster.
  - For example, the cluster's centroid, $\boldsymbol{x0}$, radius, *R*, and diameter, *D*, are

$$\boldsymbol{x}_0 = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i}{n} = \frac{LS}{n}, \quad R = \sqrt{\frac{\sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x}_0)^2}{n}} = \sqrt{\frac{nSS - 2LS^2 + nLS}{n^2}}, \quad D = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x}_j)^2}{n(n-1)}} = \sqrt{\frac{2nSS - 2LS^2}{n(n-1)}}.$$

Here, *R* is the average distance from member objects to the centroid, and *D* is the average pairwise distance within a cluster

Advantages:

1. BIRCH is efficient and scalable for large datasets, as it constructs a compact summary of the data distribution using the CF-tree.
2. It can handle high-dimensional data and incremental updates to the dataset.

Disadvantages:

1. BIRCH may not perform as well as other clustering algorithms for datasets with irregular shapes or varying densities.

2. It requires tuning of parameters such as the branching factor (B) and threshold (T), which can affect the quality of clustering results.

DBSCAN

Density Based Clustering

Density-based approach:

- Based on connectivity and density functions.
- The strategy used is to model the dense clusters separated by sparse clusters.
- The general idea is to continue growing a given cluster as long as the density (number of objects or data points) in the "neighborhood" exceeds some threshold.
- Such a method can be used to filter out noise or outliers and discover clusters of arbitrary shape.
  - Major features:
    - Discover clusters of arbitrary shape
    - Handle noise
    - One scan
    - Need density parameters as termination condition

Typical methods: DBSCAN, OPTICS, DenClue

DBSCAN

- **DBSCAN algorithm requires two parameters:**

1. **eps** : It defines the neighborhood around a data point
   - if the distance between two points is <= 'eps' then they are considered as neighbors.
   - If the eps value is chosen too small then large part of the data will be considered as outliers.
   - If it is chosen very large then the clusters will merge and majority of the data points will be in the same clusters.
   - One way to find the eps value is based on the ***k-distance graph***.

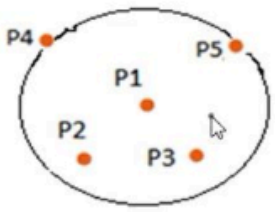
2. **MinPts**: Minimum number of neighbors (data points) within eps radius.
   - Larger the dataset, the larger value of MinPts must be chosen.
   - As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1.
   - The minimum value of MinPts must be chosen as at least 3.

DBSCAN divide data points into three types

1) Core Point 2) Noise Point 3) Border Point

1) Core Point

P1:Centre point. Radius of the circle is 2 unit.



MinPts:4 MinPts:5

P1 can form a cluster. [P2,P3,P4,P5]

P1 is core point.

MinPts:6

P1 cannot form a cluster. [P2,P3,P4,P5]. Not core point

[P1 mat be Border Point/Noise Point]

Condition to form core point

2) Noise Point

P1:Centre point, Radius of the circle is 2 unit.

MinPts:4

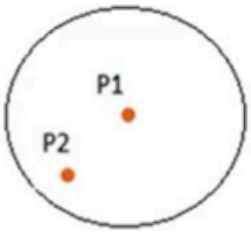
P1 is noise.

As it has only two points in its cluster.

Condition to form noise point

p is noise point

if {q |dist(p, q)<=Eps}<MinPts



In DBSCAN algorithm, first separates **core point** and **noise point**.

3) Border Point

Concept of Directly density reachable

Point q is directly density reachable from p if

dist(p, q)<=Eps

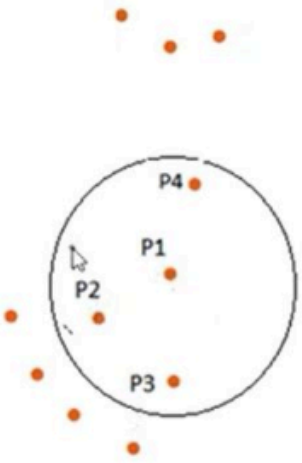
and

{r |dist(p, r)<=Eps}>=MinPts

p must be core point and q is within Eps limit from p then q is directly density reachable from p.

P2,P3 and P4 are directly density reachable from P1.

MinPts=4 Eps=2 Unit



P1 is core. P2 and P3 also act as core as various neighbours are close to them

But with P4 only P1 is closed. In P4 cluster, only two points P1, and P4 . Therefore it is not core point. P4 is noise.

Advantages:

DBSCAN is effective at identifying clusters of arbitrary shapes and handling noise and outliers in the data.

It does not require the user to specify the number of clusters beforehand, making it suitable for datasets with unknown or varying cluster structures.

Disadvantages:

DBSCAN may struggle with datasets of varying densities or with clusters of significantly different sizes.

It is sensitive to the choice of parameters (epsilon and minPts) and may require careful tuning for optimal results.

Suppose that the data mining task is to cluster points (with (x, y) representing location) into three clusters, where the points are: A_1 (2, 10), A_2 (2, 5), A_3 (8, 4), B_1 (5, 8), B_2 (7, 5), B_3 (6, 4), C_1 (1, 2), C_2 (4, 9).
The distance function is Euclidean distance. Suppose initially we assign A_1, B_1, and C_1 as the center of each cluster, respectively. Use the *k-means* algorithm to show only (i) The three cluster centers after the first round of execution (ii) The final three clusters.

Differentiate between simple linkage, average linkage and complete linkage algorithms. Use complete linkage algorithm to find the clusters from the following dataset.

X	4	8	15	24	24
Y	4	4	8	4	12

Module 5

Association Mining:
Find all frequent item sets using Apriori algorithm. List all the strong association rules.
How to compute confidence measure for an association rule?
Consider the transaction database given below. Set minimum support count as 2 and minimum confidence threshold as 70%. Generate strong association rule

Transaction ID	List of Item_Ids
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1,I3
T800	I1,I2,I3,I5
T900	I1,I2,I3

How to compute confidence for an association rule X ◊ Y?
Find all frequent item sets using Apriori algorithm. List all the strong association rules.
The transaction details are given in the following table, what is the confidence and support of the association rule {Diapers} ⇒ {Coffee, Nuts}?Find all frequent itemsets using Apriori algorithm. List all the strong association rules.

T_id	Items bought
10	Beer, Nuts, Diapers
20	Beer, Coffee, Diapers, Nuts
30	Beer, Diapers, Eggs
40	Beer, Nuts, Eggs, Milk
50	Nuts, Coffee, Diapers, Eggs, Milk

1) Suppose we have data on a few individuals randomly surveyed. The data gives the responses towards interests to promotional offers made in the areas of Finance, Travel, Reading, and Health. Sex is the output attribute to be predicted. Apply Naïve Bayesian classification algorithm to classify the new instance (Finance = No,Travel = Yes, Reading = Yes, Health = No).

2) Build Decision Tree from Following Dataset where Sex is target/Output attribute,

Finance	Travel	Reading	Health	Sex
Yes	No	Yes	No	Male
Yes	Yes	No	No	Male
No	Yes	Yes	Yes	Female
No	Yes	No	Yes	Male
Yes	Yes	Yes	Yes	Female
No	No	Yes	No	Female
Yes	No	No	No	Male
Yes	Yes	No	No	Male
No	No	No	Yes	Female
Yes	No	No	No	Male

The following table shows the midterm and final exam grades obtained for students in a database course.

x(Mid-term Exam)	Y(Final Exam)
72	84
50	63
81	77
74	78
94	90
86	75
59	49
83	79
65	77
33	52
88	74
81	90

Use the method of least squares to find an equation for the prediction of a student's final exam grade based on the student's midterm grade in the course.

Predict the final exam grade of a student who received 86 marks on the midterm exam with the abo

Write a short note on support and confidence
Explain market basket analysis with an example

- Market basket analysis is a data mining technique used by retailers to uncover associations between products purchased by customers. It works by analyzing the items that tend to be bought together in a single transaction or shopping "basket."
- The goal is to identify patterns and relationships in customer purchasing behavior to optimize strategies such as product placement, cross-selling, and promotions.
- The most common method for conducting market basket analysis is through the use of algorithms like Apriori or FP-Growth. These algorithms generate association rules, which are statements that describe the likelihood of one product being purchased given that another product is purchased.

Imagine you own a grocery store, and you want to understand the purchasing patterns of your customers. You collect transaction data, which includes the items purchased together in each transaction. Here's a small sample of your transaction data:

Transaction 1: Bread, Milk, Eggs

Transaction 2: Bread, Butter, Cheese

Transaction 3: Milk, Butter, Eggs

Transaction 4: Bread, Milk

Transaction 5: Bread, Eggs, Cheese

Using market basket analysis, you can uncover associations between items in these transactions. Let's say you set a minimum support threshold of 20%, meaning you're interested in finding associations that occur in at least 20% of the transactions.

1. Identify frequent itemsets: First, you identify the items that occur frequently enough to be considered for analysis. In this case, all items occur at least twice, so you proceed with all items.
2. Generate association rules: Next, you generate association rules to uncover relationships between items. One common algorithm for this is the Apriori algorithm.

Let's assume a simple rule: {Bread, Butter} => {Cheese}

This rule suggests that if a customer buys bread and butter, they are likely to also buy cheese.

3. Calculate support and confidence: Support measures how frequently an itemset appears in the transactions, while confidence measures the reliability of the association rule.

Support:

- Support(Bread, Butter, Cheese) = Number of transactions containing (Bread, Butter, Cheese) / Total number of transactions
- Support(Bread, Butter, Cheese) = 1 (Transaction 2)

Confidence:

- Confidence({Bread, Butter} => {Cheese}) = Support(Bread, Butter, Cheese) / Support(Bread, Butter)
- Confidence({Bread, Butter} => {Cheese}) = 1 / (Number of transactions containing (Bread, Butter) / Total number of transactions)
- Confidence({Bread, Butter} => {Cheese}) = 1 / (2 / 5) = 0.5

This means that 50% of the transactions that contain bread and butter also contain cheese.

4. Interpretation: With this rule and its associated support and confidence values, you can infer that there is a significant association between buying bread and butter together and also buying cheese. Based on this insight, you might decide to place cheese near bread and butter in your store or run promotions offering discounts on cheese when customers purchase bread and butter together.

How can we further improve the efficiency of apriori-based mining?

Improving the efficiency of Apriori

a) Hash Based Techniques

b) Transaction Reduction

c) Partitioning

d) Dynamic Itemset Counting

e) Sampling

- A hash-based technique can be used to reduce the size of the candidate k-itemsets, Ck, for k > 1.
- For example, when scanning each transaction in the database to generate the frequent 1-itemsets (L1), we can do following:
    - Generate all the 2-itemsets for each transaction,
    - Hash (i.e., map) them into the different buckets of a hash table structure, and
    - Increase the corresponding bucket counts.
- A 2-itemset with a corresponding bucket count in the hash table that is below the support threshold cannot be frequent and thus should

- be removed from the candidate set. Such a hash-based technique may substantially reduce the number of candidate k-itemsets examined (especially when k = 2).

TID	List of Items
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

Min. Support Count=3

C1

Itemset	Support Count
I1	6
I2	7
I3	6
I4	2
I5	2

Hash Function

Itemset	Count	Hash Function
I1, I2	4	[1*10+2] mod 7=**5**
I1, I3	4	[1*10+3] mod 7=**6**
I1, I4	1	[1*10+4] mod 7=**0**
I1, I5	2	[1*10+5] mod 7=**1**
I2, I3	4	[2*10+3] mod 7=**2**
I2, I4	2	[2*10+4] mod 7=**3**
I2, I5	2	[2*10+5] mod 7=**4**
I3, I4	0	--
I3, I5	1	[3*10+5] mod 7=**0**
I4, I5	0	--

Order of Items I1=1, I2=2, I3=3, I4=4, I5=5

H(x, y)=((Order of First)* 10+(Order of Second))mod 7

Hash Table Structure to generate L2

Bucket address	0	1	2	3	4	5	6
Bucket Count	2	2	4	2	2	4	4
Bucket Contents	{I1-I4}-1 {I3-I5}-1	{I1-I5}-2	{I2-I3}-4	{I2-I4}-2	{I2-I5}-2	{I1-I2}-4	{I1-I3}-4
L2	No	No	Yes	No	No	Yes	Yes

Advantages:

- Reduce the number of scans
- Remove the large candidates that cause high Input/output cost

b) Transaction Reduction

Transaction that does not contain any frequent k-itemsets cannot contain any frequent (k+1)-itemsets. Therefore, such a transaction can be marked or removed from further consideration

Trans.	Items
T1	I1, I2, I5
T2	I2, I3, I4
T3	I3, I4
T4	I1, I2, I3, I4

	I1	I2	I3	I4	I5
T1	1	1	0	0	1
T2	0	1	1	1	0
T3	0	0	1	1	0
T4	1	1	1	1	0

	I1	I2	I2	I4	I5
T1	1	1	0	0	1
T2	0	1	1	1	0
T3	0	0	1	1	0
T4	1	1	1	1	0

Minimum Support Count=2

	I1	I2	I2	I4
T1	1	1	0	0
T2	0	1	1	1
T3	0	0	1	1
T4	1	1	1	1

Trans.	Items
T1	I1, I2, I5
T2	I2, I3, I4
T3	I3, I4
T4	I1, I2, I3, I4

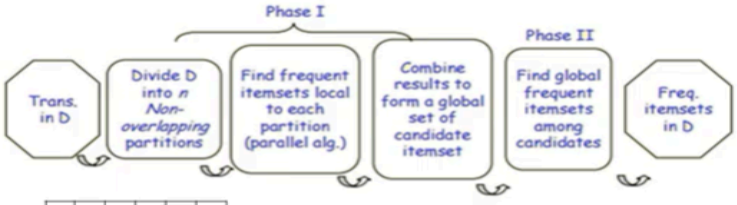
	I1,I2	I1,I3	I1,I4	I2,I3	I2,I4	I3,I4
T1	1	0	0	0	0	0
T2	0	0	0	1	1	1
T3	0	0	0	0	0	1
T4	1	1	1	1	1	1

	I1,I2	I2,I3	I2,I4	I3,I4
T2	0	1	1	1
T4	1	1	1	1

	I2,I3, I4
T2	1
T4	1

c) Partitioning

Any itemset that is potentially frequent in DB must be frequent in at least one of the partitions of DB (2 DB Scan)



	I1	I2	I3	I4	I5
T1	1	0	0	0	1
T2	0	1	0	1	0
T3	0	0	0	1	1
T4	0	1	1	0	0
T5	0	0	0	0	1
T6	0	1	1	1	0

Database is divided into three partitions.
Each having two transactions with support of 20%

Trans.	Itemset	First Scan	Second Scan	Shortlisted
		Support =20% Min. sup=1	Support =20% Min. sup=2	
T1	I1, I5	I1-1, I2-1, I4-1, I5-1	I1-1, I2-3	I2-3, I3-2
T2	I2, I4	{I1,I5}-1 {I2, I4}-1	I3-2. I4-3	I4-3 ,I5-3
T3	I4, I5	I2-1, I3-1, I4-1, I5-1	I5-3, {I1, I5}-1	{I2, I4}-2
T4	I2, I3	{I4,I5}-1,{I2, I3}-1	{I2, I4}-2, {I4, I5}-1	{I2, I3}-2
T5	I5	I2-1, I3-1,I4-1, I5-1	{I2, I3}-2, {I3,I4}-1	
T6	I2, I3, I4	{I2, I3}-1, {I2,I4}-1 {I3, I4}-1 {I2, I3, I4}-1	{I2, I3, I4}-1	

d) Dynamic Itemset Counting

It is an algorithm which reduces the number of passes made over the data while keeping the number of itemsets which are counted in any pass relatively low.

This technique can add new candidate itemsets at any marked start point of the database during the scanning of the database.

Trans.	Items
T1	A,B
T2	A
T3	B,C
T4	-

Trans.	A	B	C
T1	1	1	0
T2	1	0	0
T3	0	1	1
T4	0	0	0

Min supp = 25% and M = 2

- A dynamic itemset counting technique was proposed in which the database is partitioned into blocks marked by start points.
- In this variation, new candidate itemsets can be added at any start point, unlike in Apriori, which determines new candidate itemsets only immediately before each complete database scan.
- The technique uses the count-so-far as the lower bound of the actual count.
- If the count-so-far passes the minimum support, the itemset is added into the frequent itemset collection and can be used to generate longer candidates. This leads to fewer database scans thanwith Apriori for finding all the frequent itemsets

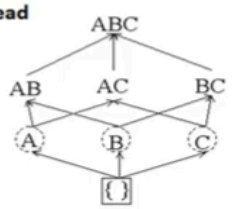
Solid box: □ Confirmed frequent itemset - an itemset we have finished counting and exceeds the support threshold *minsupp*

Solid circle: ○ Confirmed infrequent itemset - we have finished counting and it is below *minsupp*

Dashed box: ⬚ suspected frequent itemset - an itemset we are still counting that exceeds *minsupp*

Dashed circle: ◌ suspected infrequent itemset - an itemset we are still counting that is below *minsupp*

Itemset lattice before any transactions are read



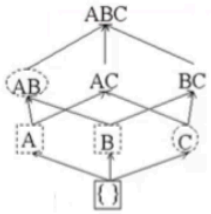
Empty itemset is marked with a **solid box.**
All 1-itemsets are marked with **dashed circles.**

Counters: A = 0, B = 0, C = 0

DIC-Algorithm

1. **Mark the empty itemset with a solid square. Mark all the 1-itemsets with dashed circles. Leave all other itemsets unmarked.**
2. While any dashed itemsets remain:
    1. Read M transactions (if we reach the end of the transaction file, continue from the beginning). For each transaction, increment the respective counters for the itemsets that appear in the transaction and are marked with dashes.
    2. If a dashed circle's count exceeds minsupp, turn it into a dashed square. If any immediate superset of it has all of its subsets as solid or dashed squares, add a new counter for it and make it a dashed circle.
    3. Once a dashed itemset has been counted through all the transactions, make it solid and stop counting it.

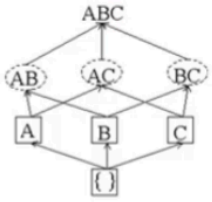
After M transactions are read



Trans.	A	B	C
T1	1	1	0
T2	1	0	0
T3	0	1	1
T4	0	0	0

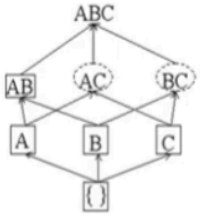
Counters: A = 2, B = 1, C = 0, AB = 0
Change A and B to **dashed boxes** because their counters are greater than minsup (1) and add a counter for AB because both of its subsets are boxes.

After 2M transactions are read



Counters: A = 2, B = 2, C = 1,
AB = 0, AC = 0, BC = 0
C changes to a square because its counter is greater than minsup.
Add counters for AC and BC because their subsets are all boxes.

After 3M transactions read

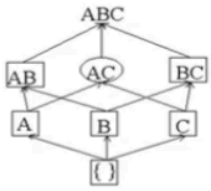


Trans.	A	B	C
T1	1	1	0
T2	1	0	0
T3	0	1	1
T4	0	0	0

Counters: A = 2, B = 2, C = 1,
AB = 1, AC = 0, BC = 0
AB has been counted all the way through and its counter satisfies minsup so we change it to a solid box. BC changes to a dashed box

https://www.youtube.com/watch?v=SLhLJZK6KaE

After 4M transactions read



Counters: A = 2, B = 2, C = 1,
AB = 1, AC = 0, BC = 1

AC and BC are counted all the way through. We do not count ABC because one of its subsets is a circle.
There are no dashed itemsets left so the algorithm is done.

e)Sampling

- The fundamental idea of the sampling approach is to select a random sample S of the given data D, and then search for frequent itemsets in S rather than D.
- It may be possible to lose the global frequent itemset. This can be reduced by lowering the minimum support.

- The sample size of S is such that the search for frequent itemsets in S can be completed in main memory, and therefore only one scan of the transactions in S is needed overall.
- Because we are searching for frequent itemsets in S rather than in D, it is possible that we will miss some of the global frequent itemsets

Explain multilevel and multidimensional association rules with example

1) Single dimensional or Intra Dimensional Association Rule

It contains a single distinct predicate (e.g. purchase) with its multiple occurrence

For e.g. purchase(X, "Milk")→ purchase(X, "Bread")

purchase(X, "Milk") ^ purchase(X, "Butter") →purchase(X, "Bread")

2) Multi dimensional or Inter Dimensional Association Rule

It contains two or more predicate. Each predicate occurs only once.

e.g.

1) Student(X, "Yes") ^ Credit Rating (X, "Excellent")→buys_Laptop(X, "Yes")

2) Student(X, "No") ^ Credit Rating (X, "Fair")→buys_Laptop(X, "No")

3) Student(X, "Yes") ^ Credit Rating (X, "Fair")→buys_Laptop(X, "No")

4) Student(X, "No") ^ Credit Rating (X, "Excellent")→buys_Laptop(X, "Yes")

For the table given, apply the Apriori algorithm and show frequent item set and strong association rules. Assume minimum support of 30% and minimum confidence of 70%

TID	Items
01	1,3,4,6
02	2,3,5,7
03	1,2,3,5,8
04	2,5,9,10
05	1,4

B) Use the Apriori algorithm to identify the frequent item-sets in the following database. Then extract the strong association rules from these sets. Assume Min. Support = 50% Min. Confidence=75%

Tid	a	b	c	d	e	f	g
Items	1,2,4,5,6	2,3,5	1,2,4,5	1,2,4,5	1,2,3,4,5,6	2,3,4	1,2,4,5

1. What do you mean by frequent itemset, frequent subsequence and frequent substructure? State one example for each.

1. Frequent item set:

- Frequent item sets, also known as association rules, are a fundamental concept in association rule mining, which is a technique used in data mining to discover relationships between items in a dataset. The goal of association rule mining is to identify relationships between items in a dataset that occur frequently together.
- A frequent item set is a set of items that occur together frequently in a dataset. The frequency of an item set is measured by the support count, which is the number of transactions or records in the dataset that contain the item set.
- For example, if a dataset contains 100 transactions and the item set {milk, bread} appears in 20 of those transactions, the support count for {milk, bread} is 20.

2. Frequent subsequence:

- Frequent pattern mining in data mining is the process of identifying patterns or associations within a dataset that occur frequently. This is typically done by analyzing large datasets to find items or sets of items that appear together frequently.
- Frequent pattern extraction is an essential mission in data mining that intends to uncover repetitive patterns or itemsets in a granted dataset. It encompasses recognizing collections of components that occur together frequently in a transactional or relational database. This procedure can offer valuable perceptions into the connections and affiliations among diverse components or features within the data.
- Frequent pattern mining has various practical uses in different domains. Some examples include market basket analysis, customer behavior analysis, web mining, bioinformatics, and network traffic analysis. Market basket analysis involves analyzing customer purchase patterns to identify connections between items and enhance sales strategies. In bioinformatics, frequent pattern mining can be used to identify common patterns in DNA sequences, protein structures, or gene expressions, leading to insights in genetics and drug design.

3. Frequent substructure:

- A frequent substructure refers to a recurring structural pattern, such as subgraphs, subtrees, or sublattices, that appears frequently in a dataset. These substructures are essential in analyzing and identifying relationships within

frequent itemsets. For instance, in the context of market basket analysis, a frequent substructure could represent a common combination of items that are frequently purchased together by customers.

- An example to illustrate this concept is as follows: Consider a dataset representing customer transactions at a supermarket. If a specific sequence of items, like purchasing milk, bread, and eggs together, occurs frequently across multiple transactions, this sequence would be considered a frequent substructure. By identifying and analyzing such frequent substructures, data miners can gain insights into patterns of customer behavior and preferences, which can be valuable for various applications like market basket analysis and targeted marketing strategies.

2. What is Market Basket Analysis ? What are the applications of market basket analysis?

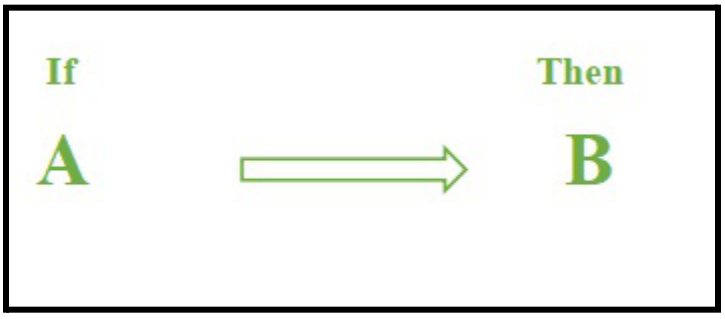
A data mining technique that is used to uncover purchase patterns in any retail setting is known as Market Basket Analysis. In simple terms Basically, Market basket analysis in data mining is to analyze the combination of products which been bought together.

This is a technique that gives the careful study of purchases done by a customer in a supermarket. This concept identifies the pattern of frequent purchase items by customers. This analysis can help to promote deals, offers, sale by the companies, and data mining techniques helps to achieve this analysis task. Example:

- Data mining concepts are in use for Sales and marketing to provide better customer service, to improve cross-selling opportunities, to increase direct mail response rates.
- Customer Retention in the form of pattern identification and prediction of likely defections is possible by Data mining.
- Risk Assessment and Fraud area also use the data-mining concept for identifying inappropriate or unusual behavior etc.

Market basket analysis mainly works with the ASSOCIATION RULE {IF} -> {THEN}.

- IF means Antecedent: An antecedent is an item found within the data
- THEN means Consequent: A consequent is an item found in combination with the antecedent.



Let's see ASSOCIATION RULE {IF} -> {THEN} rules used in Market Basket Analysis in Data Mining. For example, customers buying a domain means they definitely need extra plugins/extensions to make it easier for the users.

Like we said above Antecedent is the item sets that are available in data. By formulating from the rules means {if} component and from the example is the domain.

Same as Consequent is the item that is found with the combination of Antecedents. By formulating from the rules means {THEN} component and from the example is extra plugins/extensions.

With the help of these, we are able to predict customer behavioral patterns. From this, we are able to make certain combinations with offers that customers will probably buy those products. That will automatically increase the sales and revenue of the company.

With the help of the Apriori Algorithm, we can further classify and simplify the item sets which are frequently bought by the consumer.

There are three components in APRIORI ALGORITHM:

- SUPPORT
- CONFIDENCE
- LIFT

Applications of Market Basket Analysis :

1. Retail: Market basket research is frequently used in the retail sector to examine consumer buying patterns and inform decisions about product placement, inventory management, and pricing tactics. Retailers can utilize market basket research to identify which items are sluggish sellers and which ones are commonly bought together, and then modify their inventory management strategy accordingly.
2. E-commerce: Market basket analysis can help online merchants better understand the customer buying habits and make data-driven decisions about product recommendations and targeted advertising campaigns. The behaviour of visitors to a website can be examined using market basket analysis to pinpoint problem areas.
3. Finance: Market basket analysis can be used to evaluate investor behaviour and forecast the types of investment items that investors will likely buy in the future. The performance of investment portfolios can be enhanced by using this information to create tailored investment strategies.
4. Telecommunications: To evaluate consumer behaviour and make data-driven decisions about which goods and services to provide, the telecommunications business might employ market basket analysis. The usage of this data can enhance client happiness and the shopping experience.
5. Manufacturing: To evaluate consumer behaviour and make data-driven decisions about which products to produce and which materials to employ in the production process, the manufacturing sector might use market basket analysis. Utilizing this knowledge will increase effectiveness and cut costs.

3. Define the terms support,support count,confidence, Frequent itemset, closed frequent itemset,maximal frequent itemset with an example

- Support : It is one of the measures of interestingness. This tells about the usefulness and certainty of rules. 5% Support means total 5% of transactions in the database follow the rule.
  Support(A -> B) = Support_count(A ∪ B)
- Confidence: A confidence of 60% means that 60% of the customers who purchased a milk and bread also bought butter.
  Confidence(A -> B) = Support_count(A ∪ B) / Support_count(A)
  If a rule satisfies both minimum support and minimum confidence, it is a strong rule.
- Support_count(X): Number of transactions in which X appears. If X is A union B then it is the number of transactions in which A and B both are present.
- Maximal Itemset: An itemset is maximal frequent if none of its supersets are frequent.
- Closed Itemset: An itemset is closed if none of its immediate supersets have same support count same as Itemset.
- K- Itemset: Itemset which contains K items is a K-itemset. So it can be said that an itemset is frequent if the corresponding support count is greater than the minimum support count.
  Example On finding Frequent Itemsets – Consider the given dataset with given transactions.

TransactionId	Items
1	{A,C,D}
2	{B,C,D}
3	{A,B,C,D}
4	{B,D}
5	{A,B,C,D}

- Lets say minimum support count is 3
- Relation hold is maximal frequent => closed => frequent

1-frequent: {A} = 3; // not closed due to {A, C} and not maximal {B} = 4; // not closed due to {B, D} and no maximal {C} = 4; // not closed due to {C, D} not maximal {D} = 5; // closed item-set since not immediate super-set has same count. Not maximal

2-frequent: {A, B} = 2 // not frequent because support count < minimum support count so ignore {A, C} = 3 // not closed due to {A, C, D} {A, D} = 3 // not closed due to {A, C, D} {B, C} = 3 // not closed due to {B, C, D} {B, D} = 4 // closed but not maximal due to {B, C, D} {C, D} = 4 // closed but not maximal due to {B, C, D}

3-frequent: {A, B, C} = 2 // ignore not frequent because support count < minimum support count {A, B, D} = 2 // ignore not frequent because support count < minimum support count {A, C, D} = 3 // maximal frequent {B, C, D} = 3 // maximal frequent

4-frequent: {A, B, C, D} = 2 //ignore not frequent </

4. Explain an Apriori Algorithm for frequent itemset mining.
Apriori algorithm was found in 1994 for finding frequent itemsets in a dataset for boolean association rule. Name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties. We apply an iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

To improve the efficiency of level-wise generation of frequent itemsets, an important property is used called Apriori property which helps by reducing the search space.

Apriori Property –
All non-empty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that

All subsets of a frequent itemset must be frequent(Apriori property).
If an itemset is infrequent, all its supersets will be infrequent.

Ex:

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2
minimum confidence is 60%

Step-1: K=1
(I) Create a table containing support count of each item present in dataset – Called C1(candidate set)

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

(II) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2

- Generate candidate set C2 using L1 (this is called join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common.
- Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent.Check for each itemset)
- Now find support count of these itemsets by searching in dataset.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

- (II) compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Step-3:

- Generate candidate set C3 using L2 (join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common. So here, for L2, first element should match.
  So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}
- Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2},{I2, I3},{I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)
- find support count of these remaining itemset by searching in dataset.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

- (II) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

- Generate candidate set C4 using L3 (join step). Condition of joining Lk-1 and Lk-1 (K=4) is that, they should have (K-2) elements in common. So here, for L3, first 2 elements (items) should match.
- Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent). So no itemset in C4
- We stop here because no frequent itemsets are found further

---

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.
Confidence –
A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.
Confidence(A->B)=Support_count(A ∪ B)/Support_count(A)
So here, by taking an example of any frequent itemset, we will show the rule generation.

Itemset {I1, I2, I3} //from L3
SO rules can be
[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%
[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%
[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%
[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%
[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%
[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%
So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

5. Explain the Join and Prune step of Apriori algorithm with an example.
The Join and Prune step of the Apriori algorithm is a crucial part of the process for finding frequent itemsets. This step involves generating candidate itemsets of higher length (size) from frequent itemsets of lower length, followed by pruning those candidate itemsets that do not meet the minimum support threshold.

Here's how the Join and Prune step works with an example:

Let's consider a transactional dataset containing the following transactions:

```css
Transaction 1: {bread, milk}
Transaction 2: {bread, butter, eggs}
Transaction 3: {milk, butter, eggs}
Transaction 4: {bread, milk, butter, eggs}
```

Suppose we have a minimum support threshold of 2 (i.e., an itemset must appear in at least 2 transactions to be considered frequent).

1) Join Step:Initially, we start with frequent itemsets of length 1, which are the individual items:

```css
Frequent 1-itemsets: {bread}, {milk}, {butter}, {eggs}
```

To generate candidate itemsets of length 2, we perform a join operation on the frequent 1-itemsets. We join {bread} with {milk}, {butter}, and {eggs} to form candidate 2-itemsets:

```
Candidate 2-itemsets: {bread, milk}, {bread, butter}, {bread, eggs}
```

2) Prune Step:After generating candidate itemsets of length 2, we need to prune those that do not meet the minimum support threshold. To do this, we scan the transactional dataset to count the occurrences of each candidate itemset.

```
Count of {bread, milk} = 1
Count of {bread, butter} = 2
Count of {bread, eggs} = 1
```

Since {bread, milk} and {bread, eggs} do not meet the minimum support threshold (which is 2), we prune them from the candidate set.

```
Pruned Candidate 2-itemsets: {bread, butter}
```

3) Repeat Join and Prune Steps:We repeat the join and prune steps to generate candidate itemsets of higher length and prune those that do not meet the minimum support threshold. We continue this process until no more frequent itemsets can be generated.For example, we can generate candidate 3-itemsets by joining {bread, butter} with {milk} and {eggs}. We then prune those candidate 3-itemsets that do not meet the minimum support threshold.

```
Candidate 3-itemsets: {bread, butter, milk}, {bread, butter, eggs}
```

We prune {bread, butter, milk} since it does not meet the minimum support threshold.

```
Pruned Candidate 3-itemsets: {bread, butter, eggs}
```

Since no more candidate itemsets can be generated, we stop the process.
In summary, the Join and Prune step of the Apriori algorithm involves generating candidate itemsets by joining frequent itemsets of lower length and pruning those candidate itemsets that do not meet the minimum support threshold. This process is repeated iteratively to find all frequent itemsets in the dataset.

6. Advantages and disadvantages of Apriori Algorithm
Advantages:

1. Calculation of Large Itemsets: The primary advantage of the Apriori algorithm is its ability to efficiently calculate large itemsets. It uses a "bottom-up" approach, starting from itemsets of size 1 and iteratively finding larger itemsets by combining smaller ones. This allows it to handle datasets with a large number of transactions and items effectively.
2. Simple to Understand and Apply: Another significant advantage of Apriori is its simplicity. The algorithm's concept is straightforward and intuitive, making it

easy to understand and implement, even for those new to data mining and machine learning. This simplicity contributes to its popularity and widespread use in both academia and industry.

3. Flexible: The algorithm can be adapted to different types of data and applications, including market basket analysis, recommender systems, web usage mining, and more. It can handle both categorical and binary data, making it versatile for a wide range of use cases.
4. Provides interpretable results: The output of the Apriori algorithm, in the form of frequent item sets and association rules, is easy to interpret and understand. This makes it useful for generating actionable insights and making informed decisions in various domains.

Disadvantages:

1. Computational Expense: One of the main drawbacks of the Apriori algorithm is its computational expense, particularly in terms of calculating support. Support refers to the frequency of occurrence of an itemset in the dataset. Since Apriori needs to scan the entire database to calculate support for each itemset, it can be time-consuming and resource-intensive for large datasets.
2. Large Number of Candidate Rules: In some cases, the Apriori algorithm may generate a large number of candidate rules, especially when dealing with datasets containing a vast number of items or transactions. These candidate rules need to be evaluated to identify meaningful associations, which can significantly increase the computational complexity of the algorithm. As a result, processing such datasets can become computationally expensive and may require substantial computational resources.
3. Apriori property assumption: The algorithm relies on the Apriori property, which states that if an item set is frequent, then all of its subsets must also be frequent. While this assumption helps reduce the search space, it may not always hold true in practice, leading to potentially missed associations or false discoveries.
4. Need for threshold setting: The Apriori algorithm requires users to specify minimum support and confidence thresholds to determine which item sets and association rules are considered meaningful. Setting these thresholds appropriately can be challenging and may require domain knowledge or trial and error.

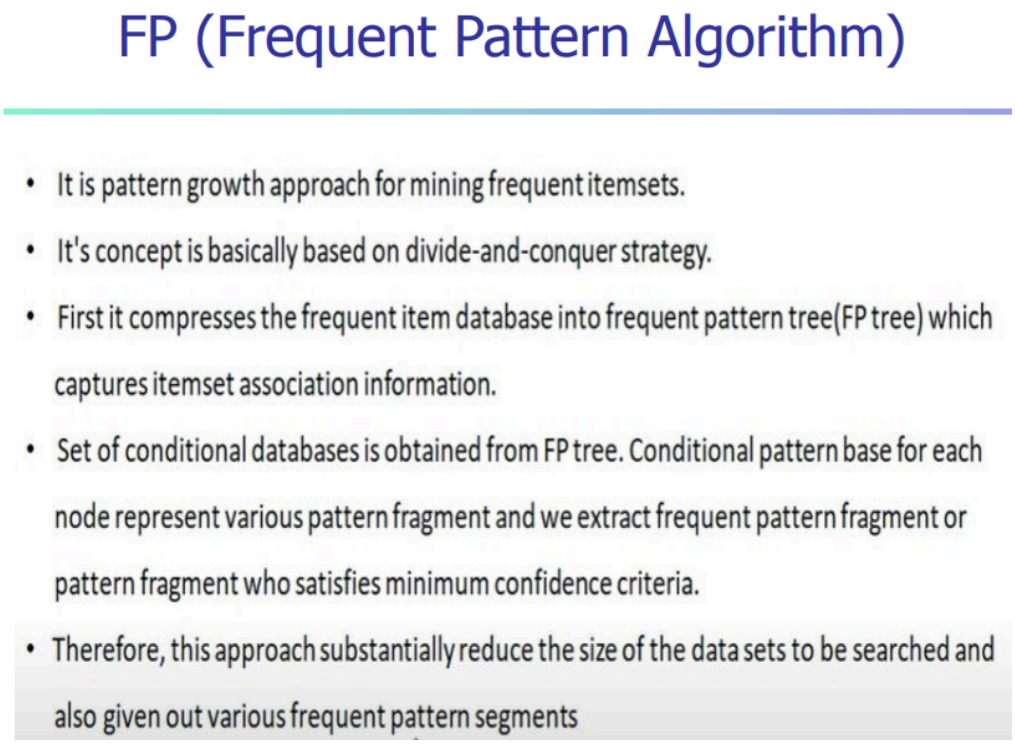
7. State applications of Apriori Algorithm

The Apriori algorithm is a popular algorithm in data mining and association rule learning. It is used to identify frequent item sets and relevant associations among items in a transactional database. Here are some applications of the Apriori algorithm:

1. **Market Basket Analysis:** One of the most common applications of the Apriori algorithm is in market basket analysis. It helps retailers understand the purchasing behavior of customers by identifying which items are frequently bought together. This information can be used for product placement, targeted marketing, and personalized recommendations.

2. **Recommender Systems:** Apriori algorithm can be used in recommender systems to suggest related items to users based on their past preferences or behavior. By identifying frequent item sets, the algorithm can recommend items that are often purchased or viewed together, improving the user experience and increasing sales.
3. **Web Usage Mining:** In web usage mining, the Apriori algorithm can be applied to analyze web log data and identify patterns in user navigation behavior. This information can be used to improve website design, optimize content placement, and enhance the effectiveness of online advertising campaigns.
4. **Healthcare Data Analysis:** In healthcare, the Apriori algorithm can be used to analyze patient records and identify patterns in medical diagnoses, treatments, and outcomes. This information can help healthcare providers make more informed decisions, improve patient care, and detect potential medical errors or fraud.
5. **Cross-Selling and Upselling:** E-commerce companies often use the Apriori algorithm to identify cross-selling and upselling opportunities. By analyzing transaction data, the algorithm can recommend complementary products or upgrades to customers based on their past purchases, increasing sales and customer satisfaction.
6. **Fraud Detection:** In banking and finance, the Apriori algorithm can be used to detect suspicious patterns or anomalies in transaction data that may indicate fraudulent activity. By identifying frequent item sets or unusual transaction sequences, the algorithm can help detect and prevent fraudulent transactions in real-time.

8. Explain Frequent pattern algorithm. State advantages of it over Apriori algorithm



FP (Frequent Pattern Algorithm)

Advantages of FP Tree

- Only 2 passes over data-set
- "Compresses" data-set
- No candidate generation
- Much faster than Apriori

Disadvantages of FP Tree

- FP-Tree may not fit in memory!!
- FP-Tree is expensive to build

Advantages of FP-Growth over Apriori:

1. Efficiency: FP-Growth is generally more efficient than the Apriori algorithm, especially for large datasets. It achieves this efficiency by compressing the dataset into a compact FP-tree structure and avoiding the generation of candidate itemsets.
2. Reduced Computational Overhead: Unlike Apriori, which generates candidate itemsets and scans the entire dataset multiple times, FP-Growth requires only two passes over the dataset: one for constructing the FP-tree and another for mining frequent itemsets. This reduces computational overhead and improves performance.
3. Less Dependency on Memory: FP-Growth typically requires less memory compared to Apriori, especially for datasets with high-dimensional itemsets. The FP-tree structure efficiently represents the frequency of itemsets without the need to store candidate itemsets explicitly.
4. No Need for Candidate Generation: Unlike Apriori, which generates candidate itemsets based on the Apriori property, FP-Growth does not require the generation of candidate itemsets. Instead, it directly constructs the FP-tree and mines frequent itemsets from it, avoiding the overhead associated with candidate generation.
5. Scalability: FP-Growth is highly scalable and can handle large datasets with millions of transactions and high-dimensional itemsets efficiently. Its efficient use of memory and reduced computational overhead make it suitable for real-world applications with big data.

Module 6

1. What is BI?BI Applications,

What is Business Intelligence?

BI(Business Intelligence) is a set of processes, architectures, and technologies that convert raw data into meaningful information that drives profitable business actions. It is a suite of software and services to transform data into actionable intelligence and knowledge.

BI has a direct impact on organization's strategic, tactical and operational business decisions. BI supports fact-based decision making using historical data rather than assumptions and gut feeling.

BI tools perform data analysis and create reports, summaries, dashboards, maps, graphs, and charts to provide users with detailed intelligence about the nature of the business.

Why is BI important?

- Measurement: creating KPI (Key Performance Indicators) based on historic data
- Identify and set benchmarks for varied processes.
- With BI systems organizations can identify market trends and spot business problems that need to be addressed.
- BI helps on data visualization that enhances the data quality and thereby the quality of decision making.
- BI systems can be used not just by enterprises but SME (Small and Medium Enterprises)

BI Applications

1. Reporting and Dashboards: These applications allow users to create and customize reports and dashboards to visualize key performance indicators (KPIs), trends, and metrics across various aspects of the business.

2. Data Visualization Tools: Data visualization tools enable users to create visual representations of data, such as charts, graphs, and maps, to better understand patterns, correlations, and outliers within the data.

3. Online Analytical Processing (OLAP): OLAP tools enable users to interactively analyze multidimensional data, such as sales data by product, region, and time, to gain deeper insights into business performance.

4. Data Mining and Predictive Analytics: These applications leverage statistical algorithms and machine learning techniques to identify patterns, trends, and relationships within data and make predictions about future outcomes.

5. ETL (Extract, Transform, Load) Tools: ETL tools facilitate the extraction, transformation, and loading of data from disparate sources into a centralized data warehouse or data lake, ensuring data consistency and reliability for analysis.

6. Performance Management Solutions: Performance management solutions help organizations track and monitor performance against predefined goals and objectives, enabling them to make data-driven decisions to improve business outcomes.

7. Mobile BI: Mobile BI applications allow users to access BI reports, dashboards, and analytics on mobile devices, enabling real-time decision-making and collaboration regardless of location.

8. Self-Service BI: Self-service BI tools empower non-technical users to independently access and analyze data without relying on IT support, enabling faster decision-making and agility within the organization.

9. Data Governance and Compliance Tools: These tools help organizations ensure data quality, integrity, and compliance with regulatory requirements by establishing policies, standards, and controls for data management.

10. Collaborative BI: Collaborative BI tools enable users to share insights, collaborate on analysis, and make collective decisions within teams or across departments, fostering a culture of data-driven collaboration within the organization.

Applications of BI

1. Financial Analytics: BI applications in finance help organizations analyze financial data to improve budgeting, forecasting, and financial performance management. They enable financial analysts to track key metrics such as revenue, expenses, profitability, and cash flow, and identify trends and anomalies for better decision-making.

2. Sales and Marketing Optimization: BI tools are used to analyze sales and marketing data to optimize strategies, improve customer segmentation, and enhance campaign effectiveness. They help organizations identify high-value customers, track sales performance, monitor market trends, and evaluate marketing ROI.

3. Supply Chain Management: BI applications in supply chain management enable organizations to optimize inventory levels, streamline logistics, and enhance supplier performance. They provide insights into demand forecasting, inventory turnover, transportation costs, and supplier relationships to improve operational efficiency and reduce costs.

4. Customer Relationship Management (CRM): BI tools integrated with CRM systems help businesses analyze customer data to gain insights into customer behavior, preferences, and satisfaction levels. They enable organizations to personalize marketing efforts, improve customer service, and increase customer retention and loyalty.

5. Human Resources Analytics: BI applications in human resources enable organizations to analyze workforce data to optimize recruitment, performance

management, and employee retention. They provide insights into employee demographics, skills, training needs, and engagement levels to support strategic workforce planning and talent management.

6. Healthcare Analytics: BI tools in healthcare help providers and payers analyze clinical, operational, and financial data to improve patient care, reduce costs, and enhance operational efficiency. They enable healthcare organizations to track patient outcomes, identify healthcare trends, and optimize resource allocation.

7. Retail Analytics: BI applications in retail help retailers analyze sales data, customer behavior, and inventory levels to optimize merchandising, pricing, and promotional strategies. They provide insights into product performance, customer segmentation, and market trends to drive sales and improve profitability.

8. Manufacturing Analytics: BI tools in manufacturing enable organizations to analyze production data, equipment performance, and supply chain operations to optimize manufacturing processes and improve product quality. They provide insights into production efficiency, maintenance scheduling, and demand forecasting to reduce downtime and increase productivity.

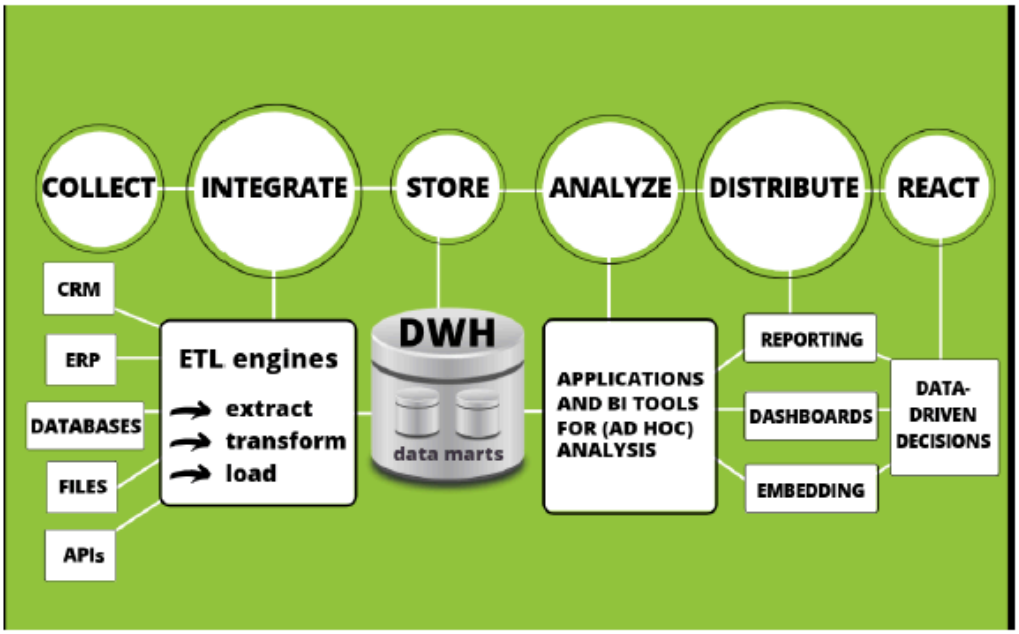
9. Risk Management and Compliance: BI applications help organizations analyze risk exposure, regulatory compliance, and fraud detection across various business functions. They provide insights into potential risks, regulatory requirements, and compliance issues to mitigate risks and ensure adherence to legal and industry standards.

10. Executive Decision Support: BI applications provide senior executives with real-time access to key performance indicators and strategic insights to support decision-making. They enable executives to monitor business performance, identify opportunities and threats, and align organizational strategies with business objectives.

2. Business intelligence architectures;

A solid BI architecture framework consists of:

1. Collection of data
2. Data integration
3. Storage of data
4. Data analysis
5. Distribution of data
6. Reaction based on insights



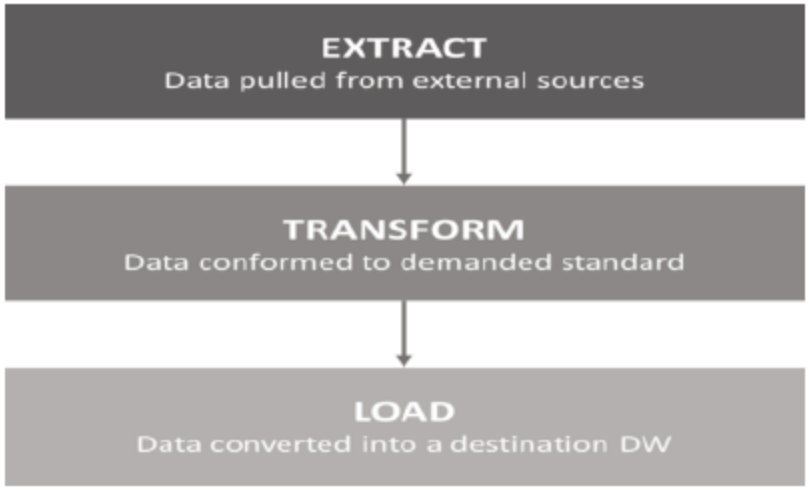
In above BI architecture diagram, we can see how the process flows through various layers, and now we will focus on each.

1. Collection of data

The first step in creating a stable architecture starts in gathering data from various data sources such as CRM, ERP, databases, files or APIs, depending on the requirements and resources of a company. Modern BI tools offer a lot of different, fast and easy data connectors to make this process smooth and easy by using smart ETL engines in the background. They enable communication between scattered departments and systems that would otherwise stay disparate. From a business point of view, this is a crucial element in creating a successful data-driven decision culture that can eliminate errors, increase productivity, and streamline operations. You have to collect data in order to be able to manipulate with it.

2. Data integration

When data is collected through scattered systems, the next step continues in extracting data and loading it to a data warehouse. This process is called ETL (Extract-Transform-Load).



With an increasing amount of data generated today and the overload on IT departments and professionals, ETL as a service comes as a natural answer to solve complex data requests in various industries. The process is simple; data is pulled from external sources (from our step 1) while ensuring that these sources aren't negatively impacted with the performance or other issues. Secondly, data is conformed to the demanded standard. In other words, this (transform) step ensures data is clean and prepared to the final stage: loading into a data warehouse.

3. Data storage
Store data in DWH.

4. Analysis of data
In this step of our compact BI architecture, we will focus on the analysis of data after it's handled, processed, and cleaned in former steps with the help of data warehouse(s). The ubiquitous need for successful analysis for empowering businesses of all sizes to grow and profit is done through BI application tools. Especially when it comes to ad hoc analysis that enables freedom, usability, and
flexibility in performing analysis and helping answer critical business questions swiftly and accurately.

This visual above represents the power of a modern, easy-to-use BI user interface. Modern BI tools empower business users to create queries via drag and drop, and build stunning data visualizations with a few clicks, even without profound technological knowledge. This simplifies the process of creating business dashboards, or an analytical report, and generate actionable insights needed for improving the operational and strategic efficiency of a business. The data warehouse works behind this process and makes the overall architecture possible.

5. Data distribution
Data distribution comes as one of the most important processes when it comes to sharing information and providing stakeholders with indispensable insights to obtain sustainable business development. Distribution is usually performed in 3 ways:

a) Reporting via automated e-mails: Created reports can be shared with selected recipients on a defined schedule. The dashboards will be automatically updated on a daily, weekly or monthly basis which eliminates manual work and enables up to date information.
b) Dashboarding: Another reporting option is to directly share a dashboard in a secure viewer environment. The users you share with cannot make edits or change the content but can use assigned filters to manipulate data and interact with the dashboard. Another option is to share via public URL that enables users to access the dashboards even if they're outside of your organization, as shown in the picture below:
c) Embedding: This form of data distribution is enabled through embedded BI. Your own application can use dashboards as a mean of analytics and reporting without the need for labelling the BI tool in external applications or intranets.

6. Reactions based on generated insights
The final stage where the BI architecture expounds its power is the fundamental part of any business: creating data-driven decisions. Without the backbones of data warehousing and business intelligence, the final stage wouldn't be possible and businesses won't be able to progress. CEOs, managers, professionals, coworkers, and all the interested stakeholders can have the power of data to generate valid, accurate, data-based decisions that will help them move forward. Let's see this through one of our dashboard examples: the management KPI dashboard.

This dashboard is the final product on how data warehouse and business intelligence work together. The processes behind this visualization include the whole architecture which we have described, but it would not be possible to achieve without a firm data warehouse solution. Ultimately, this enables a high-level manager to get a comprehension of the strategic development and potential decisions for creating and maintaining a stable business. On this particular dashboard, you can see the total revenue, as well as on a customer level, adding also the costs. The targets are also set so that the dashboard immediately calculates if they have been met or additional adjustments are needed from a management point of view. As revenue is one of the most important factors when evaluating if the business is growing, this management dashboard ensures all the essential data is visualized and the user can easily interact with each section, on a continual basis, making the decision processes more cohesive and, ultimately, more profitable.

3. Development of a business intelligence system using Data Mining for business Applications like Fraud Detection, Recommendation, Retail etc.
Developing a business intelligence system leveraging data mining techniques for various business applications like fraud detection, recommendation systems, and retail analytics involves several key steps and considerations:

1. Identify Business Objectives: Clearly define the business goals and objectives that the business intelligence system aims to support. For example, reducing fraudulent

activities, improving customer satisfaction through personalized recommendations, or optimizing retail operations.

2. Data Collection and Integration: Gather relevant data from various sources such as transactional data, customer data, product data, etc. Integrate the data into a centralized data warehouse or data lake, ensuring data quality and consistency.

3. Data Preprocessing: Cleanse, transform, and preprocess the data to make it suitable for analysis. This may involve handling missing values, outlier detection, normalization, and feature engineering.

4. Algorithm Selection: Choose appropriate data mining algorithms and techniques based on the specific business requirements and objectives. For fraud detection, algorithms like anomaly detection, classification, and clustering may be used. For recommendation systems, collaborative filtering, content-based filtering, or hybrid approaches can be employed. Retail analytics may involve techniques such as market basket analysis, customer segmentation, and predictive modeling.

5. Model Training and Evaluation: Train machine learning models using historical data and evaluate their performance using metrics relevant to the business objectives. This may involve splitting the data into training and testing sets, cross-validation, and fine-tuning model parameters.

6. Integration with Business Processes: Integrate the developed models and insights into existing business processes and decision-making workflows. This could involve building dashboards, reports, or real-time alerting systems to enable stakeholders to act upon the generated insights.

7. Continuous Monitoring and Improvement: Regularly monitor the performance of the deployed models and update them as needed to adapt to changing business conditions and evolving data patterns. This may involve retraining models with new data, incorporating feedback from users, and refining algorithms.

8. Data Privacy and Security: Ensure compliance with data privacy regulations and implement robust security measures to protect sensitive business information. This is particularly important when dealing with customer data and financial transactions.

9. Scalability and Flexibility: Design the business intelligence system to be scalable and flexible enough to accommodate growing data volumes and evolving business requirements over time. This may involve leveraging cloud-based solutions, distributed computing frameworks, and modular architectures.

10. User Training and Adoption: Provide training and support to users to ensure they can effectively utilize the business intelligence system to make informed decisions.

Encourage user adoption by demonstrating the value of the insights generated by the system.

3. What are the components of BI architecture

Business intelligence refers to a collection of mathematical models and analysis methods that utilize data to produce valuable information and insight for making important decisions.

Main Components of Business Intelligence System:
Data Source
Data Mart / Data Warehouse
Data Exploration
Data Mining
Optimization
Decisions
1.Data Source:

To begin, the first step is gathering and consolidating data from an array of primary and secondary sources. These sources vary in origin and format, consisting mainly of operational system data but also potentially containing unstructured documents like emails and data from external providers.

2.Data Mart / Data Warehouse:

Through the utilization of extraction and transformation tools, also known as extract, transform, load (ETL), data is acquired from various sources and saved in databases designed specifically for business intelligence analysis. These databases, commonly known as data warehouses and data marts, serve as a centralized location for the gathered data.

3.Data Exploration:

The third level of the pyramid offers essential resources for conducting a passive analysis in business intelligence. These resources include query and reporting systems, along with statistical methods. These techniques are referred to as passive because decision makers must first develop ideas or establish criteria for data extraction before utilizing analysis tools to uncover answers and confirm their initial theories. For example, a sales manager might observe a decrease in revenues in a particular geographic region for a specific demographic of customers. In response, she could utilize extraction and visualization tools to confirm her hypothesis and then use statistical testing to validate her findings based on the data.

4.Data Mining:

The fourth level, known as active business intelligence methodologies, focuses on extracting valuable information and knowledge from data. Part II of this book will delve into various techniques such as mathematical models, pattern recognition, machine learning, and data mining. Unlike the tools discussed in the previous level, active models do not rely on decision makers to come up with hypothesis but instead aim to enhance their understanding.

5.Optimization:

As you ascend the pyramid, you'll encounter optimization models that empower you to choose the most optimal course of action among various alternatives, which can often be quite extensive or even endless. These models have also been effectively incorporated in marketing and logistics.

6.Decisions:

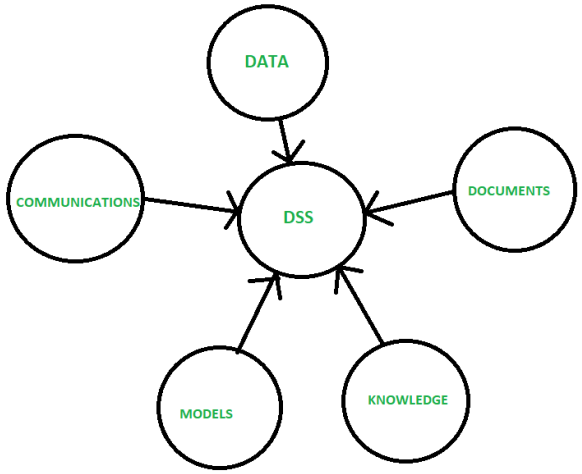
At last, the pinnacle of the pyramid reflects the ultimate decision made and put into action, serving as the logical end to the decision-making process. Despite the availability and effective utilization of business intelligence methodologies, the decision still lies in the hands of the decision makers, who can incorporate informal and unstructured information to fine-tune and revise the suggestions and outcomes generated by mathematical models.

4. What is a decision support system (DSS)?Examples

Decision Support System (DSS): It's a computer-based system that aids the process of decision-making. It is an interactive, flexible and adaptable computer system. It is specially developed for supporting the solution of a non-structured management problem for improved decision-making. DSS is a specific class of computerized information systems that supports business and organizational decision-making activities.

Components of DSS:

Model Management
Data Management
User Interface Management



Key components of a DSS typically include:
Database Management System (DBMS): Stores and manages the data used by the DSS.
Model Base: Contains mathematical and statistical models used for analysis and decision-making.
User Interface: Provides a user-friendly interface for interacting with the DSS, including data entry, querying, and visualization.
Knowledge Base: Stores domain-specific knowledge, rules, and guidelines used in decision-making.
Decision Support Software: Includes analytical tools, reporting capabilities, and visualization techniques for generating insights and recommendations.

5. What are the characteristics of DSS?

- Support for decision-makers in semi-structured and unstructured problems.
- Support for managers at various managerial levels, ranging from top executive to line managers.
- Support for individuals and groups. Less structured problems often requires the involvement of several individuals from different departments and organization level.
- Support for interdependent or sequential decisions.
- Support for intelligence, design, choice, and implementation.
- Support for variety of decision processes and styles.
- DSSs are adaptive over time.

6. What are the advantages and disadvantages of DSS.
Advantages :
It saves time.
Enhances efficiency.
Reduces the cost.
It improves personal efficiency.
It increases the decision maker satisfaction.

Disadvantages :

Information Overload.
Status reduction.
Over-emphasize decision making.

7. What are the types of DSS?

Types of Decision Support systems are Document-driven, Data-driven, Knowledge-driven, Model-driven, and Communication-driven.

Data-driven DSS

A data-driven DSS is a computer program that makes decisions based on data from internal databases or external databases. Typically, a data-driven DSS uses data mining techniques to discern trends and patterns, enabling it to predict future events. Businesses often use data-driven DSSes to help make decisions about inventory, sales and other business processes. Some are used to help make decisions in the public sector, such as predicting the likelihood of future criminal behavior.

Model-driven DSS

Built on an underlying decision model, model-driven decision support systems are customized according to a predefined set of user requirements to help analyze different scenarios that meet these requirements. For example, a model-driven DSS may assist with scheduling or developing financial statements.

Communication-driven and group DSS

A communication-driven and group decision support system uses a variety of communication tools -- such as email, instant messaging or voice chat -- to allow more than one person to work on the same task. The goal behind this type of DSS is to increase collaboration between the users and the system and to improve the overall efficiency and effectiveness of the system.

Knowledge-driven DSS

In this type of decision support system, the data that drives the system resides in a knowledge base that is continuously updated and maintained by a knowledge management system. A knowledge-driven DSS provides information to users that is consistent with a company's business processes and knowledge.

Document-driven DSS

A document-driven DSS is a type of information management system that uses documents to retrieve data. Document-driven DSSes enable users to search webpages or databases, or find specific search terms. Examples of documents accessed by a document-driven DSS include policies and procedures, meeting minutes and corporate records.