Unit 1:

Introduction to Data Warehousing & OLAP

Data Warehousing According to BillInmon:

"Data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision-making process."

Difference between OLTP and OLAP

- **OLTP** stands for: **Online Transactional Processing**
- **OLAP** stands for: **Online Analytical Processing**

S.No	Features	OLTP	OLAP
1	Characteristics	Operational processing	Informational processing
2	Orientation	Transaction	Analysis
3	Users	Clerk, DBA, Database professional	College worker (e.g., manager, executive, analyst)
4	Function	Day-to-day operation	Long-term, informational requirements, decision support
5	DB Design	ER-based, applicationoriented	Star / Snowflake, subject-oriented
6	Summarization	Primitive, highly detailed	Summarized, consolidated
7	Data	Current; guaranteed up-todate	Historical; accuracy maintained over time
8	View	Detailed, flat relational	Summarized, multidimensional
9	Unit of Work	Short, simple transaction	Complex query
10	Access	Read / Write	Mostly Read
11	Focus	Data in	Information out

12	Operations	Indexed based on primary keys	Lots of scans

13	Number of Records Accessed	Tens	Millions
14	No. of Users	Thousands	Hundreds
15	DB Size	Hundreds MB to GB	Hundreds GB to TB
16	Priority	High performance, high availability	High flexibility, end-user autonomy
17	Metric	Transaction throughput	Query throughput

Applications of Data Warehousing

1. Banking

- Stores customers' transaction data from various branches.
- Detects fraud and unusual transactions.
- Helps in loan processing, risk analysis, and financial reporting.
- Improves decision-making for new services or branches.

2. Telecommunication

- Manages huge volumes of call records and customer data.
- Analyzes network usage and performance.
- Helps in billing, offers planning, and reducing customer churn.
- Supports marketing and customer segmentation.

3. Retail & Inventory

- Tracks product sales, customer buying behavior, and stock levels.
- Helps in demand forecasting and seasonal trend analysis.
- Improves supply chain efficiency.
- Plans promotions, discounts, and product placement.

4. Hospital / Healthcare

- Stores patient records, treatment history, and test results.
- Analyzes disease patterns, doctor performance, and resource usage.
- Helps in improving patient care and hospital management.

- Used in insurance verification and billing systems.

5. **Agriculture**
   - Helps in better planning and farming.
   - Predicts best time to grow crops based on weather and past data.
   - Helps use the right tools at the right time, which improves smart farming.
   - Helps farmers to track market prices to sell at the right time.
6. **Education**
   - Stores student records, marks, and attendance.
   - Analyzes exam performance and progress.
   - Helps plan curriculum and teaching methods.
   - Generates reports for parents and management.
7. **Manufacturing**
   - Tracks production and machine performance.
   - Monitors quality control and defects.
   - Helps in cost reduction and efficiency.
   - Manages raw materials and inventory levels.

Data Warehouse Architecture

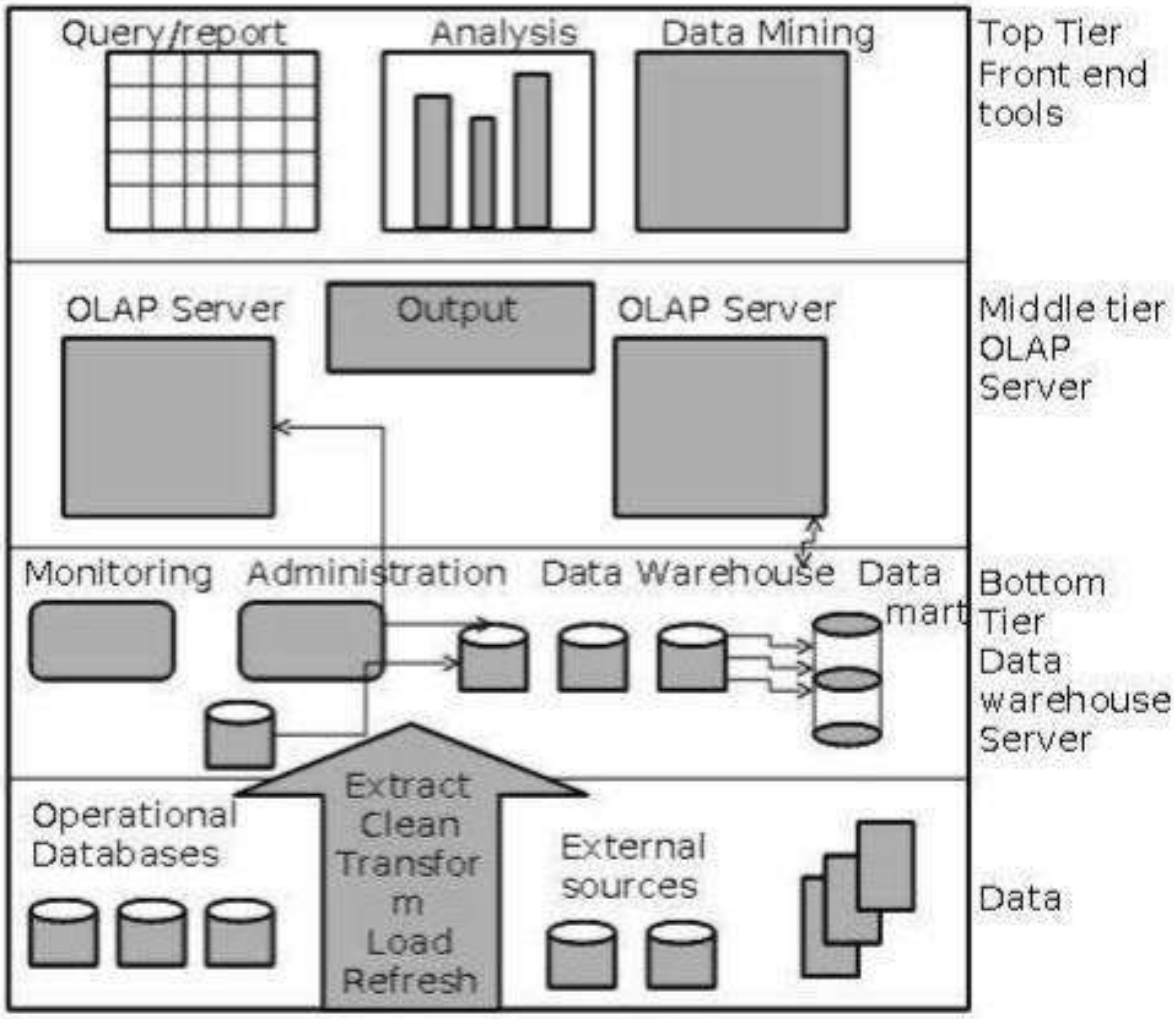
Definition:

Data warehousing architecture refers to the overall design and structure that supports the data warehouse system, helping organizations store, manage, and analyze large amounts of data. It ensures:

- **Security**
- **Scalability**
- **Accessibility** of data for **analysis** and **decision-making**.

Main Components of Data Warehousing Architecture:

1. **Top Tier:** Front-end tools
2. **Middle Tier:** OLAP servers
3. **Bottom Tier:** Data warehouse server



1. Top Tier: Front End Tools

The top tier is the user interface layer of the data warehouse. It includes tools and applications that allow end-users (like business analysts, managers, data scientists) to interact with the processed data.

The purpose of the top tier is to enable users to view reports, run queries, perform analysis, visualize trends and discover hidden patterns.

Components:

1. Query / Report
2. Analysis
3. Data Mining

1. Query / Report

✱ It is used to retrieve specific data from the data warehouse.

✱ Helps users to generate reports in a readable format (tables, summaries).

✱ Often involve SQL queries or drag-and-drop report builders.

✱ Reports can be scheduled (automated) or customized (on-demand).

✱ Data can be filtered, sorted, and exported.

How it works:

- User inputs a query (e.g., through SQL or drag and drop interface).
- The tool sends the query to the OLAP servers or data warehouse.
- The system fetches the data from the warehouse.
- The result is formatted into a report (table, list, PDF, Excel, etc).
- Reports can be saved, printed or shared.

Uses:

- Generate sales or attendance reports
- View/export data in readable form.

2. ### Analysis:

- Used for interactive and visual data analysis
- Allow users to drill-down, roll-up, slice and dice data (OLAP operations)
- Data is shown in charts, graphs, dashboards and pivot tables
- Helps in identifying trends and comparing performance

How it works:

- Users select dimensions (e.g., Time, Region, Product) to view.
- The OLAP server builds a data cube for fast multi-dimensional analysis
- Users can perform OLAP operations
- Results are displayed in dashboards or graphs.

Uses:

- Compare performance visually
- Spot trends and patterns quickly

3. ### Data Mining:

- ✶ It applies machine learning and statistical techniques to discover patterns.
- ✶ Helps in prediction, classification, clustering and association
- ✶ Useful for future planning and decision-making
- ✶ Extract hidden insights from large datasets.

How it works:

✶ User selects a dataset from the data warehouse

✶ The system trains the model using historical data

✶ It then generates insights, rules or forecasts

✶ Results are shown as text, charts or rules.

Uses:

✶ Predict customer behavior

✶ Discover sales trends

✶ Detect fraud or anomalies

2. **Middle Tier: OLAP Server:**

The Middle Tier is the OLAP (Online Analytical Processing) server, which sits between the Data Warehouse (Bottom Tier) and the Front-end Tools (Top Tier).

- It is responsible for processing queries, managing multidimensional data (Cubes), and enabling fast, interactive analysis.

OLAP Server:

→ OLAP stands for Online Analytical Processing

→ An OLAP server is a middle layer in the data warehouse architecture that is used to:

✶ Organize data into multi-dimensional cubes

✶ Quickly answer complex analytical queries

✶ Support decision-making with summarized and detailed views of data

How it works:

1. **User Request:**

A user (via a front-end tool) requests data like:

"Show sales for January 2025 by Region and Product"

2. **Fetch Data:**

The **OLAP server** fetches preprocessed data from the data warehouse.

3. **Create / Access Data Cube:**

Data is organized in a **multidimensional cube** (e.g., Time, Product, Region).

4. **Perform OLAP operations:**

- **Slice:** View a single layer (e.g., Jan only)
- **Dice:** View a sub-cube (e.g., Phones in Asia in Jan & Feb)

- **Roll-Up:** Summarize (e.g., Monthly → Quarterly sales)
- **Drill-Down:** Go into detail (e.g., Region → City → Store)

5. Send Results:

Sends the processed result to the front-end tool for the user to view (table/chart)

Uses:

★ It makes it easy to view data from different perspectives.

★ It supplies processed data for reports, graphs, and dashboards.

3. Bottom-Tier: Data Warehouse Server

The bottom tier is the foundation of the data warehouse architecture.

It includes the data warehouse servers, where all raw data is **collected, cleaned, transformed**, and **stored** from multiple sources.

- It is responsible for managing huge volumes of data.
- It is responsible for storing historical and current data in an organized way.

1. Monitoring:

Monitoring refers to tracking the performance and health of the data warehouse system.

How it works:

- Continuously checks system metrics like data load time, storage usage and query speed.
- Detects failures, slowdowns, or errors in ETL processes or server performance.

Uses:

- Alerts if ETL or server fails.
- Helps maintain fast performance for users.

2. Administration:

It manages and controls the data warehouse system and its users.

How it works:

- Sets user roles and permissions (who can access what).
- Handles data backups, recovery, and security.
- Schedules ETL jobs and updates.

Uses:

- Ensures data availability and integrity.
- Automates routine tasks.

3. Metadata Repository:

A library of information about the data (data about data). **How it works:** → Stores:

- Data source (e.g., "from sales DB")
- Format (e.g., date, currency)
- Load time (e.g., updated every day

Uses:

- Maintains consistency and traceability
- Helps users find and understand the meaning of data

4. Data Warehouse: Definition:

A central repository that stores large volumes of cleaned, transformed and integrated data from multiple sources.

How it works:

Receives data through the ETL process

- Organizes data into tables and relationships
- Keeps historical and current data for analysis Uses:
- Supports OLAP analysis and reporting
- Provides a single source of truth for the entire organization

5. Data Marts:

Definition:

A smaller, focused part of the data warehouse built for specific business units (e.g., HR, sales, finance)

How it works:

- Extracts a subset of relevant data from the main warehouse
- Can be created quickly for fast access Uses:
- Provides faster performance for department-level analysis
- Simplifies data access for non-technical users

4. Data:

- Data is a collection of raw facts and figures.
- It could be numbers, text, dates, or images used to represent information.

1. Operational Databases: Definition:

Operational databases are real-time databases used in daily business operations.

How it works:

- They record live transactions and activities.
- Data is extracted from these systems regularly (daily / weekly)
- Then it goes through **ETL**
- Final clean data is stored in the data warehouse for analysis

Uses:

- Acts as a main source of raw data
- Captures all current business transactions

2. **External Sources:**

Definition:

External sources are data sources outside the organization.

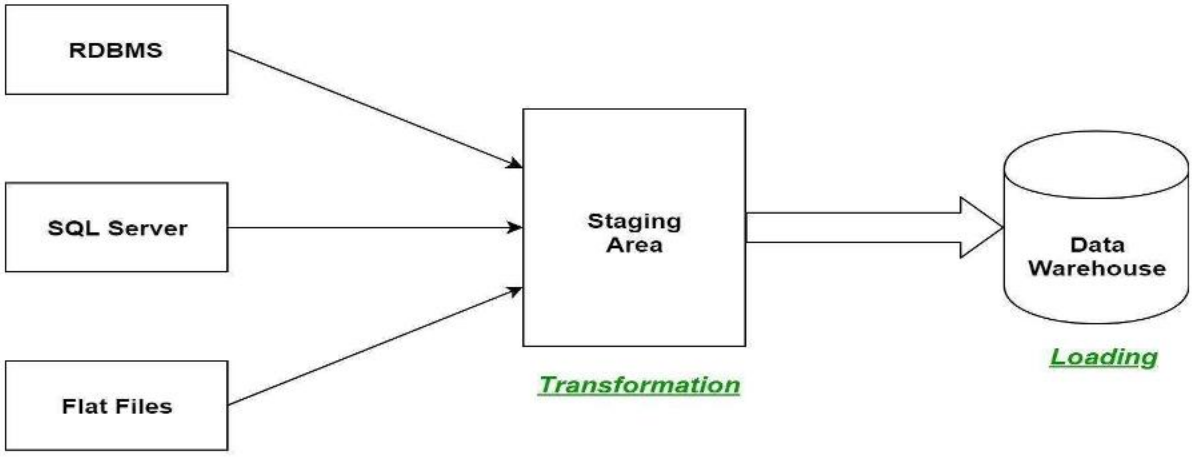
Examples:

Government statistics

- Suppliers / vendors
- **Cloud data How it works:**
- External data is extracted via APIs, file uploads or downloads
- Goes through data cleaning and transformation
- **Integrated into the data warehouse Uses:**
- Helps in market research, trend analysis, competitor tracking
- Complements internal data for better decision-making

ETL Process: Extraction, Transformation, Loading

The ETL process, which stands for **Extraction, Transformation, and Loading**, is a critical methodology used to prepare data for storage, analysis, and reporting in a data warehouse. It involves three distinct stages that help to streamline raw data from multiple sources into a clean, structured, and usable form.



1. Extraction

The **Extract** phase is the first step in the ETL process, where raw data is collected from various data sources. These sources can be diverse, ranging from:

- **Structured sources** like databases (SQL, NoSQL)
- **Semi-structured data** like JSON, XML
- **Unstructured data** such as emails

or flat files The main goal of **extraction** is to **gather data**.

Without altering its format, enabling it to be further processed in the next stage.

Types of data sources can include:

- **Structured**: SQL databases, ERPs, CRMs
- **Semi-structured**: JSON, XML
- **Unstructured**: Emails, web pages, flat files

2. Transformation:

The **transform phase** is where the magic happens. Data extracted in the previous phase is often **raw** and **inconsistent**.

During transformation, the data is **cleaned**, **aggregated**, and **formatted** according to business rules. This is a crucial step because it ensures that the data meets the **quality standards** required for accurate analysis.

Common transformations include:

- **Data Filtering**: Removing irrelevant or incorrect data
- **Data Sorting**: Organizing data into a required order for easier analysis
- **Data Aggregating**: Summarizing data to provide meaningful insights (e.g., averaging sales data)

The transformation stage can also involve more complex operations such as:

- **Currency conversions**
- **Text normalization**
- Or applying **domain-specific rules** to ensure the data aligns with **organizational needs.**

Staging Area:

The **staging area** is a temporary location used during the ETL (Extraction, Transformation, Loading) process.

It acts as a **buffer zone** where raw data from multiple sources is collected, cleaned, and prepared before being loaded into the data warehouse.

What kind of data is stored in the staging area?

- Raw extracted data
- Intermediate transformation results
- Error logs or rejected data

Note: Data is **not kept permanently**.

Once data is transformed and loaded into the warehouse, it may be **deleted or archived**. It's only used by **ETL developers and systems**.

1. Extraction:

i) SQL Server

- A **SQL Server** is a **Relational Database Management System (RDBMS)** developed by **Microsoft**.
- Used to **store, manage, and retrieve** data efficiently
- **Popular in large enterprise environments Key Points:**
- Supports **large databases** and **complex queries**
- Can handle **transaction processing, analytics, and reporting**
- Ensures **data security, backup, and recovery**
- **Supports integration with cloud Used In:**
- Banking systems
- E-commerce websites
- Company HR and payroll systems

What kind of data is stored in the staging area?

- Raw extracted data
- Intermediate transformation results
- Error logs or rejected data

Note: Data is **not kept permanently**.

Once data is transformed and loaded into the warehouse, it may be **deleted or archived**. It's only used by **ETL developers and systems**.

2. Extraction:

i) SQL Server

- A **SQL Server** is a **Relational Database Management System (RDBMS)** developed by **Microsoft**.
- Used to **store, manage, and retrieve** data efficiently
- **Popular in large enterprise environments Key Points:**
- Supports **large databases** and **complex queries**
- Can handle **transaction processing, analytics, and reporting**
- Ensures **data security, backup, and recovery**
- **Supports integration with cloud Used In:**
- Banking systems
- E-commerce websites
- Company HR and payroll systems

3. Flat Files

- A simple text file that stores data **without structured relationships**. **Common formats**:

CSV (.csv)

TXT (.txt)

TSV (.tsv)

Key Points:

- No table structure, primary keys, or constraints
- Easy to create and share between systems
- Used for: data export, logs, backups, and data exchange

Used In:

- Exporting sales reports
- System or application logs
- Sensor or IoT data files

4. RDBMS (Relational Database Management System)

- A software system used to store structured data in tables **with relationships**.
- Follows **relational model** (rows = records, columns = fields) **Key Points:**
- Data is organized using tables, rows, columns, and keys
- Ensures data consistency, integrity, and normalization
- Supports SQL to manage and retrieve data
- **Includes features like:**
  - Constraints
  - Indexes
  - Joins
  - Triggers

- **Used in:**
  - Banking system
  - School / college databases
  - E-commerce platforms.

3. Loading:

Once data has been cleaned and transformed, it is ready for the final step: **Loading**. This phase involves transferring the transformed data into a data warehouse, data lake, or another target system for storage. Depending on the use case, there are two types of loading methods:

1. Initial / Full Load:

- All data is loaded into the target system, often used during the initial population of the warehouse.
- Old data is deleted or replaced.
- Simple to implement but not efficient for large datasets.
- Useful when the source data is small or updated rarely.

2. Incremental / Incremental Load:

- Only new or updated data is loaded, making this method more efficient for ongoing data updates.

- Faster and more efficient than full load.
- It is used in systems where data changes frequently.
- Reduces load time, especially in large datasets.
- Can be scheduled daily | hourly or in real-time.

OLAP: Multi-Dimensional Data Model

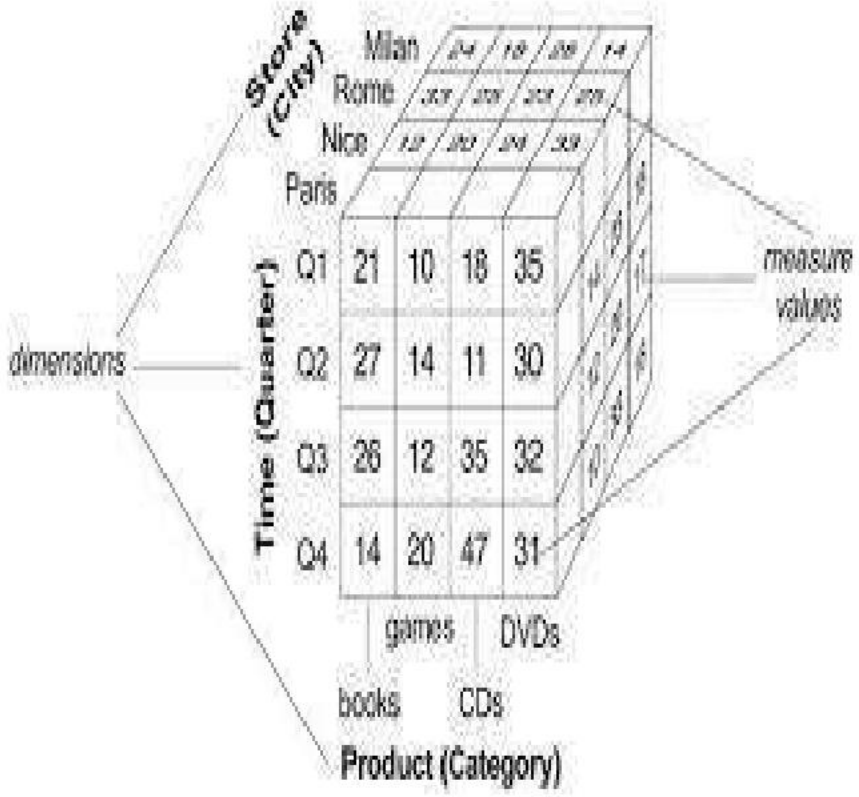
Data warehouses and OLAP tools are based on a multi-dimensional data model. The model views data in the form of a data cube.

Data cube: A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.

Each dimension may have a table associated with it. It's called a *dimension table*, which further describes the dimension.

Designed for complex queries like trends, summaries, comparisons, etc.

(Diagram of a data cube with dimensions: Location, Time, and Item)



1. **Fact Table:**

Contains quantitative data (measurable) like sales, revenue, profit.

2. **Dimension Table:**

Contains descriptive data (who, what, when, where, how) like product, time, location.

Advantages:

- ✶ Easy to understand for users
- ✶ Fast query performance
- ✶ Helps in trend analysis, forecasting, decision making.

Hierarchy

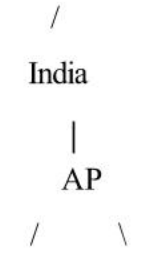
A **hierarchy** represents levels of data organization within a dimension. It allows users to view data at different levels — from high-level summaries to detailed views.

- A structure within a dimension that defines **parent-child relationships**.
- Used to **navigate and analyze** data across levels.
- Helps in **grouping**, **filtering**, and **aggregating** data easily.
- Improves **query performance** by organizing data logically.

Examples:

Ex (1): Location Hierarchy

markdown World

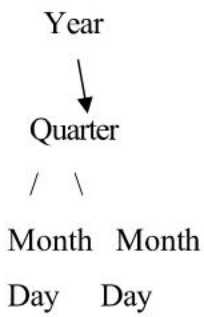


- Hierarchy path: World → Country → State → City

Ex(2):TimeHierarch

sql

CopyEdit



- Hierarchy path: Year → Quarter → Month → Day

Lattice - Cube

A **Lattice Cube** (also called **Cube Lattice**) in a data warehouse is a conceptual structure that represents all possible combinations of aggregations (called **cuboids**) for a set of dimensions in a data cube.

- ✶ It is a **directed graph** showing all levels of summarization, from the most detailed (**base cuboid**) to the most summarized (**apex cuboid**).
- ✶ Each **node** is a **cuboid** (a data cube with a specific combination of dimensions).
- ✶ Used in **OLAP** to **precompute and store** different aggregation levels for faster query performance.
- ✶ It helps in **materializing only useful cuboids**.
- ✶ **Reduces computation time** and **query response time** in OLAP.
- ✶ Used in **data cube computation algorithms** like star-cubing, etc.

Diagram: Lattice Cube Example

sql CopyEdit

```
      Year
    /    \

  Quarter  Week
   /   \
 Month  Days
```

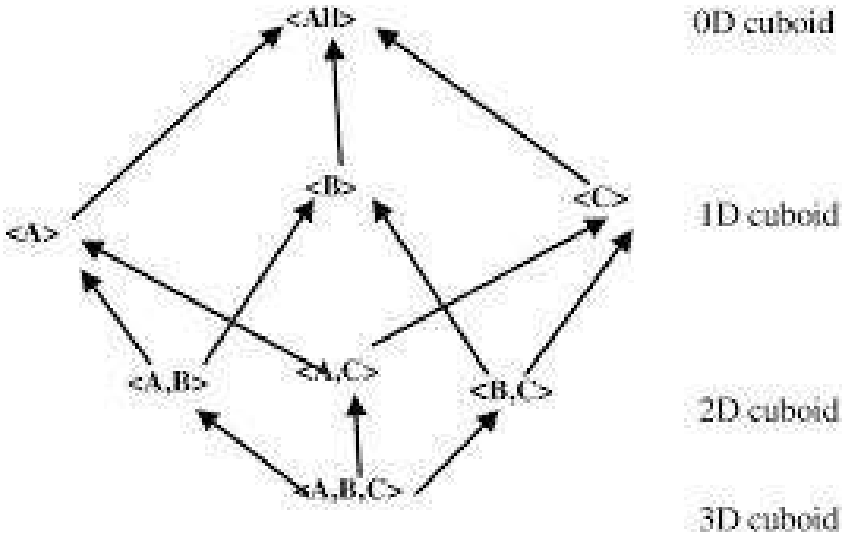
Cuboid:

A **cuboid** is a multi-dimensional view of data for a specific combination of dimensions.

- ✶ The **cuboid** that holds the **lowest level of summarization** is called the **base cuboid**.
- ✶ The **apex cuboid**, or **0-D cuboid**, refers to the case where the **group-by is empty**. It contains the total sum of all sales.

  The **apex cuboid** is the **most generalized (least specific)** of the cuboids and is often denoted as **"all"**.

Diagram Explanation:



Types of Cuboids:

1. **Apex Cuboid:**
   - Top-most: **all**
   - ✶ This is the **apex cuboid (0-D)**.
   - ✶ No dimensions – just one big total (e.g., total sales of everything).
2. **1-D Cuboids:**

   These use only one dimension:
   - Supplier
   - Time
   - Item
   - Location
3. **2-D Cuboids:**

These combine two dimensions, like:
- Time, Time
- Item, location
- Location,supplier

4. **3-D Cuboids:**

These use three dimensions, like:
- Time, Item, Supplier
- Item, Location, Supplier etc.

5. **Base Cuboid:**

- It contains full detailed data with all dimensions.

How It Works:

- You can move **up** from bottom to top to get **summarized data** (*roll-up*).
- You can move **down** to get **more detailed data** (*drill-down*).
- This helps answer **different types of queries quickly** in **OLAP systems**. **OLAP Operations:**

OLAP operations are:

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

1. SLICE:

✶ The slice operation selects data for a single value of one dimension and removes that dimension from the cube.

✶ This results in a reduced-dimensional view (i.e., a smaller cube).

✶ It analyzes data by selecting a specific subset of data based on one or more dimensions.

✶ It involves filtering the data cube to focus on a particular combination of dimension values.

✶ The slice operation creates a sub-set of data from the original multidimensional dataset.

✶ This subset contains only the data that matches the selected dimension values.

✶ Slicing allows users to focus their analysis on a particular aspect of the data.

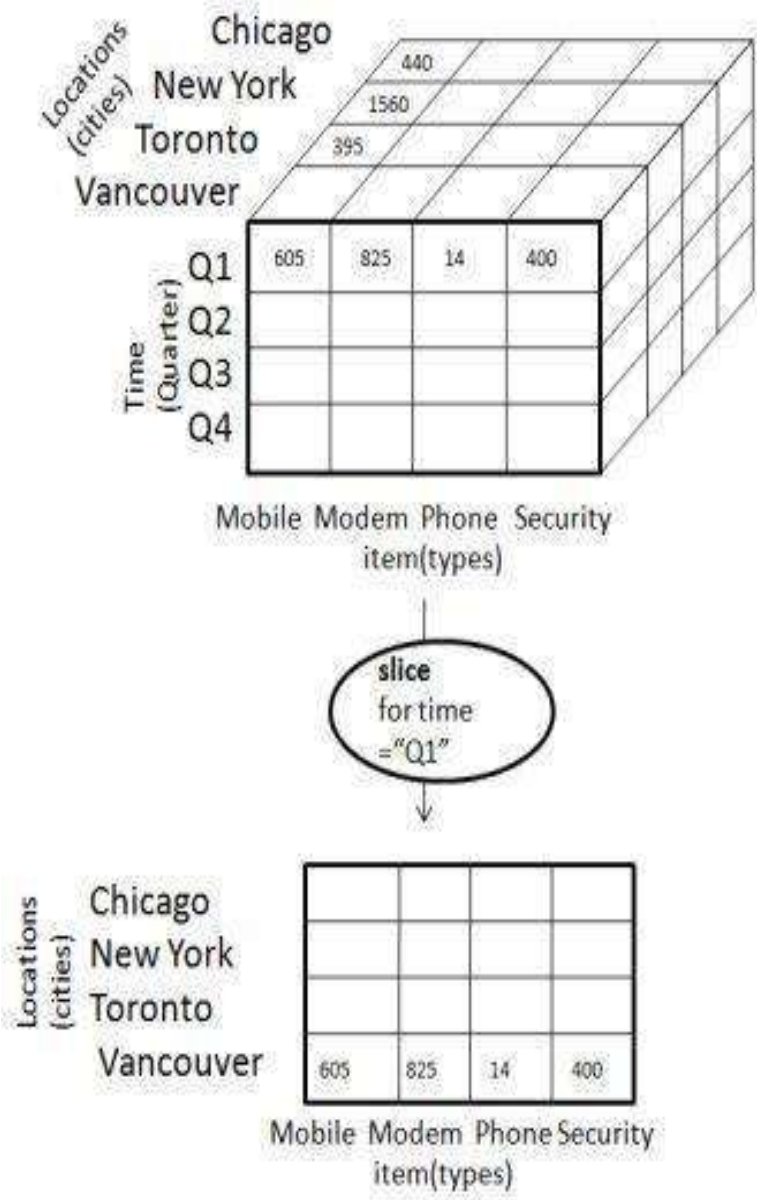
Example:

A 3D cube diagram is shown with the following axes:

- **Location (cities):** Chicago, New York, Toronto, Vancouver
- **Time (Quarters):** Q1, Q2, Q3, Q4
- **Item (types):** home entertainment, computer, phone, security Values shown in the cube (partial): Q1: 605, 825, 14, 400

Note:

performing all OLAP operations using this example.

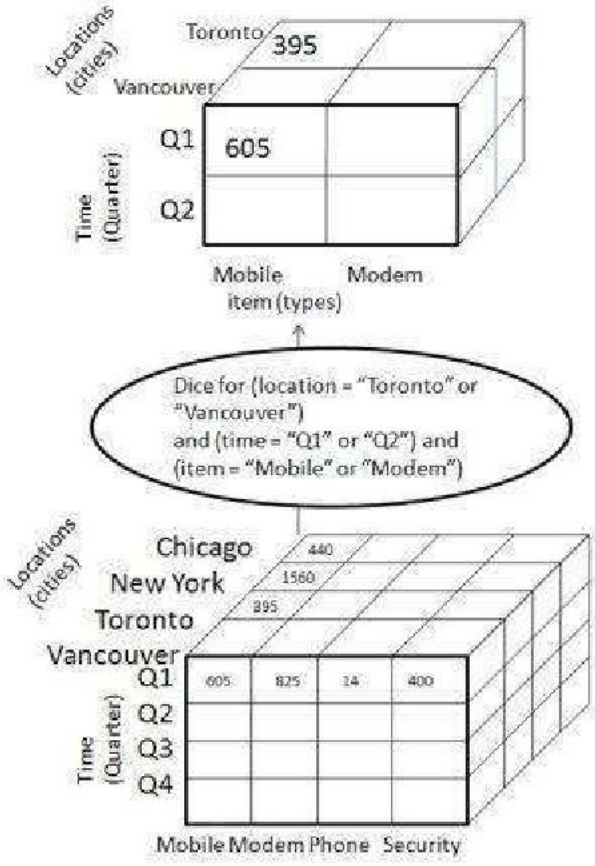


2. **Dice:**

Dice is an OLAP operation that allows users to analyze data by selecting a specific combination of dimension values to create a subcube.

This operation enables users to view data from multiple perspectives simultaneously, providing a focused and tailored view of the data.

The dicing operation creates a smaller, more targeted subcube from the original multidimensional data cube.



3. Roll-Up:

Roll-up is an **OLAP operation** that allows users to navigate from a lower level of detail to a higher level of summary within a dimension.

- This operation helps users view broader trends and patterns in data and gain insights at higher levels of aggregation.

Users can navigate up the hierarchy to view data at broader, more summarized levels.

- This supports **strategic decision-making** and planning based on broader trends.
- When roll-up is performed, one or more dimensions from the data cube are removed.
- Roll-up performs **aggregation** on a data cube in any of the following ways:

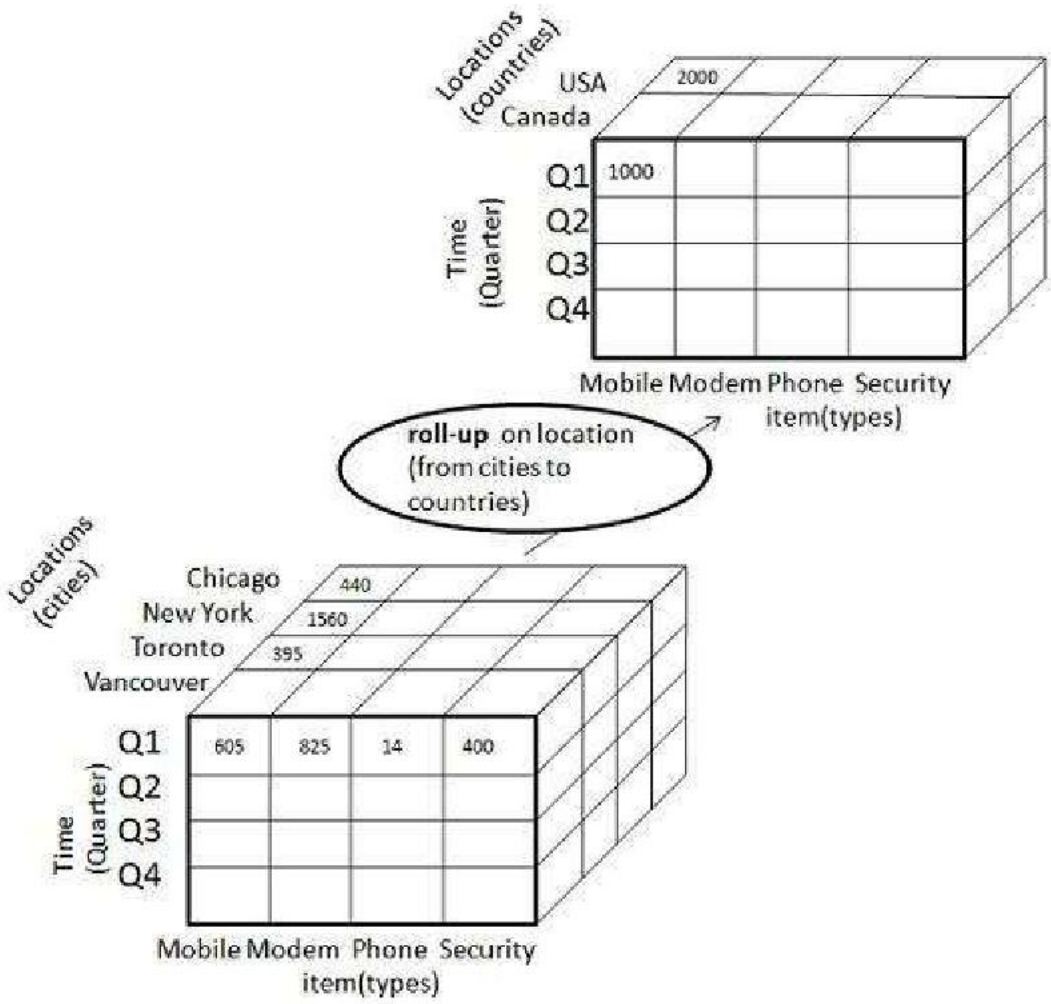
o By climbing up a concept hierarchy for a dimension o

By dimension reduction

- Roll-up is performed by climbing up a **concept hierarchy** for the dimension (e.g., location).
- On rolling up, the data is aggregated by ascending the **location hierarchy**

— from the level of **city** to the level of **country**.

Roll up:



4. **Drill-Down**

- **Drill-Down** is an OLAP operation that allows users to go from summarized data to more detailed data.
- It's like zooming in on your data.
- You move from higher-level to lower-level data within a dimension hierarchy.

Hierarchical Navigation: Requires a dimension with a defined hierarchy.

Increases Detail: Each step adds more specific info.

Interactive: Often done via dashboard tools (click to drill). **Reversible:** Can combine with Roll-up to go back up.

Improves Clarity: Helps pinpoint issues or opportunities.

Business Intelligence Tools: Tableau, Power BI, etc.

✶ It ensures your data model has proper hierarchies.

✶ Use visual tools (charts, tables) for cleaner drill-downs.

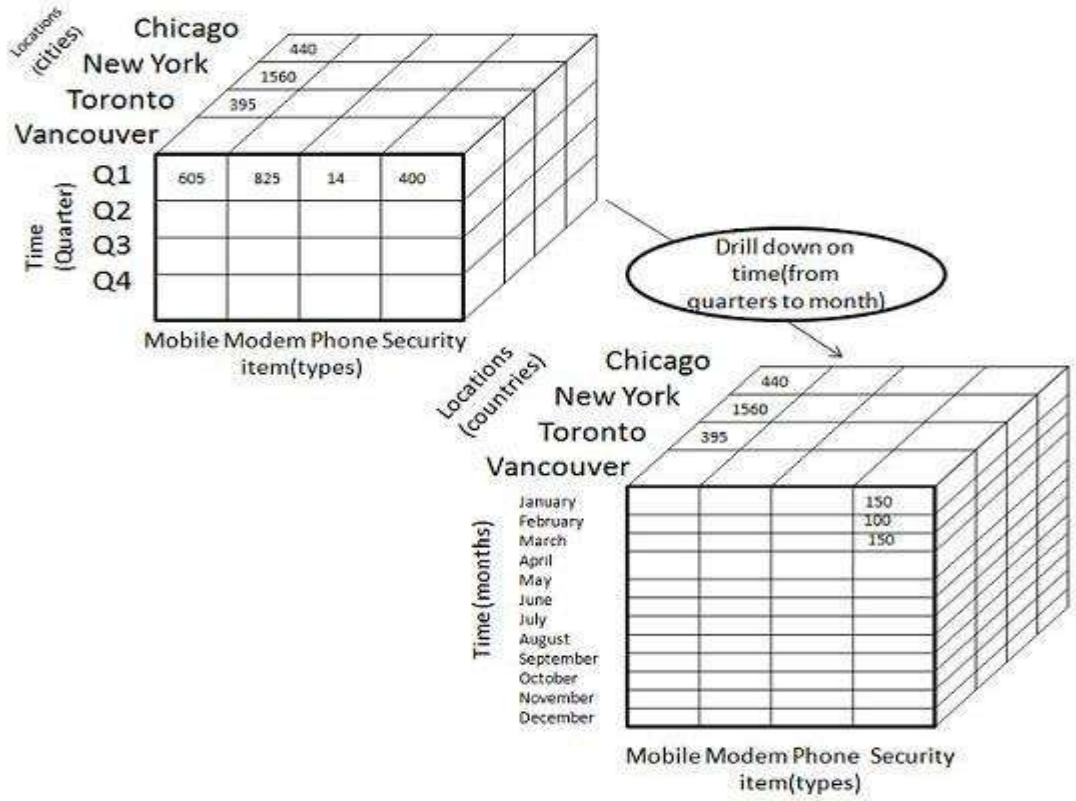
✶ Avoid too much drilling - it can get overwhelming.

✶ Combine with filters (slicing/dicing) for precise analysis.

✶ Improves decision-making by showing detailed facts.

✶ Requires that data has a defined hierarchy.

✶ Saves time by showing only the needed details without changing the whole report.



5. Pivot:

✶ Pivot means rotating the view of data to see it from a different angle.

✶ It changes the perspective of the data by rearranging dimensions.

✶ It doesn’t change the data, just how you view it.

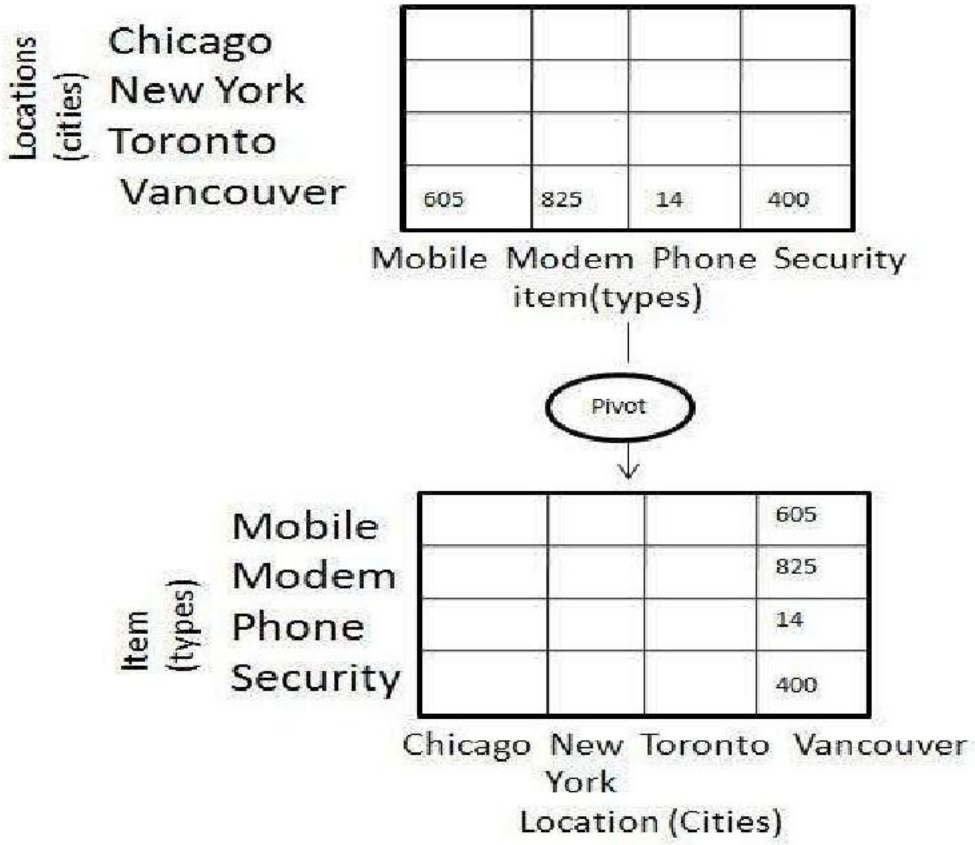
✶ Pivot is also called Rotate in OLAP terminology.

✶ It changes the orientation of the data view.

✶ Pivot operation does not change the actual data, only how it's displayed.

✶ Used to make comparisons easier.

For example:



Star Schema:

- The **star schema** is a data modeling technique used in data warehouses to organize data for fast querying and analysis.
- It is called a star because the diagram looks like a star with a central table and surrounding related tables.

✶ It is simple and easy to understand.

✶ It is best suited for OLAP operations.

✶ Ideal for reporting and business analysis.

✶ It is fast for sized heavy operations.

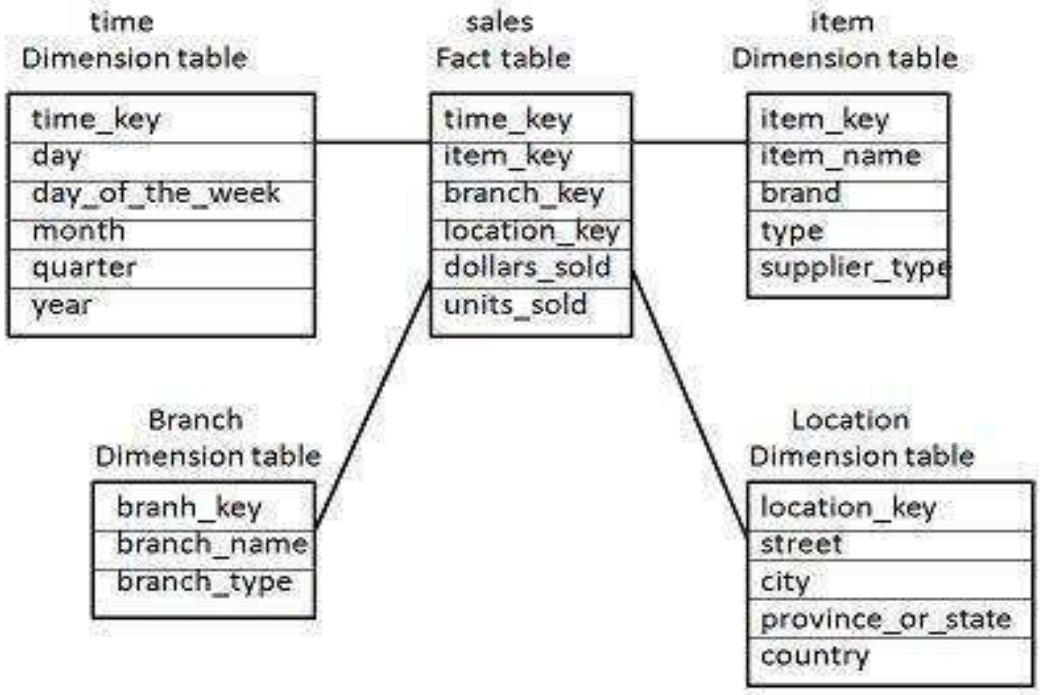
✶ Easy for business users to understand.

✶ Helps in aggregating data quickly.

✶ Takes up more storage due to data repetition (**denormalization**).

✶ Each dimension in a star schema is represented with only one dimension table.

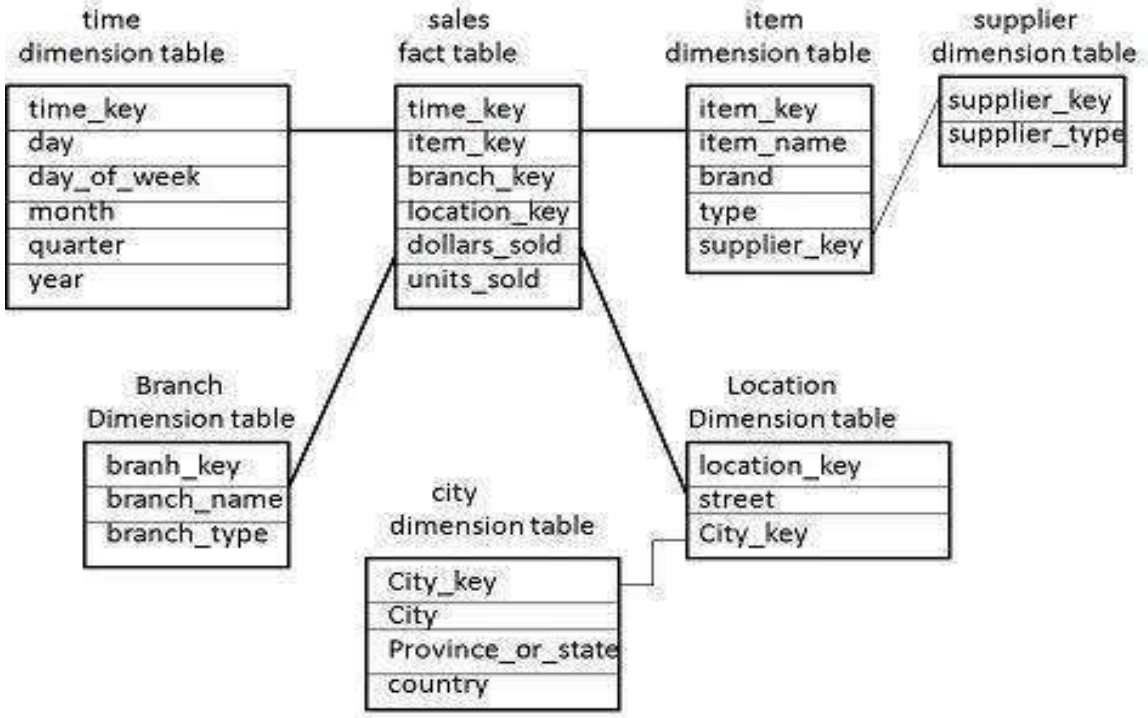
Table Structures in the Star Schema:



Snowflake Schema:

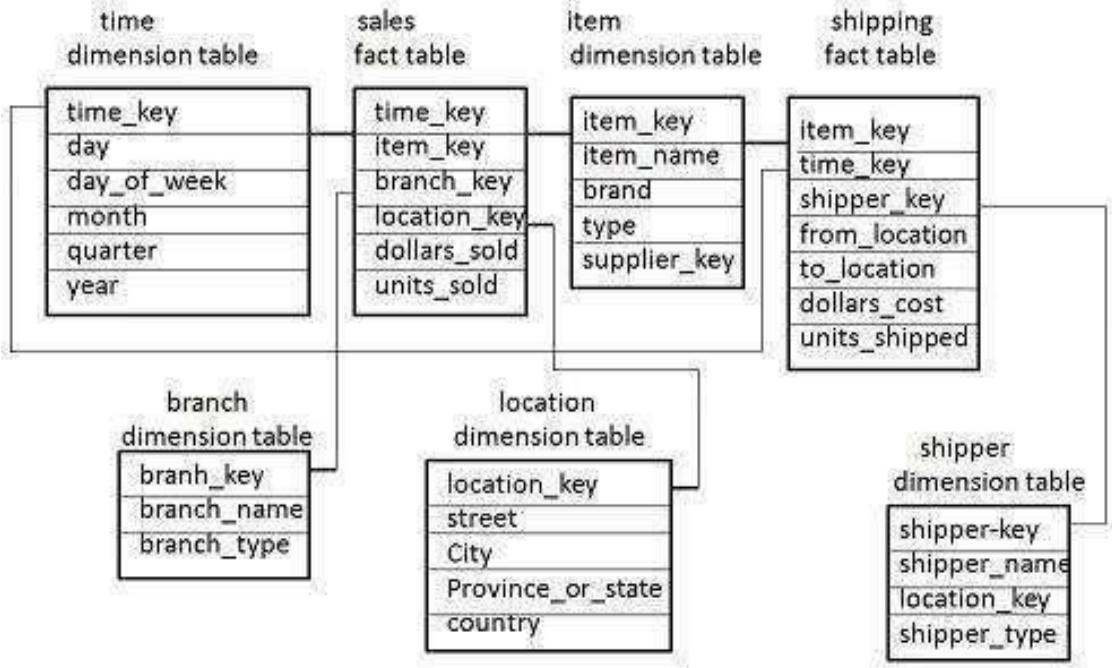
Some dimension tables in the Snowflake schema are normalized. The normalization splits up the data into additional tables.

Unlike star schema, the dimension table in a snowflake schema are normalized. For example, the item dimension table in star schema is normalized and split into two dimension tables, namely item and supplier table.



3: Fact Constellation Schema

A fact constellation has multiple fact tables. It is also known as **galaxy schema**. The following diagram shows two fact tables, namely **sales** and **shipping**.



The sales fact table is same as that in the star schema.

The shipping fact table has the five dimensions, namely item_key, time key, shipper_key, from_location, to_location.

The shipping fact table also contains two measures, namely dollars_sold and units_sold.

It is also possible to share dimension tables between fact tables. For example, time, item and location dimension tables are shared between the sales and shipping fact table.

UNIT 2

DATA PRE-PROCESSING AND DATA UNDERSTANDING

DATA PRE-PROCESSING:

Data preprocessing is the **initial and essential phase** in the data mining process, where raw data is transformed into a clean, consistent, and analyzable format before applying mining algorithms.

It involves a collection of **data preparation techniques** that handle problems like **missing values, noisy data, inconsistencies, irrelevant attributes, and different data formats** to improve the quality of the dataset.

In real-world scenarios, raw data collected from various sources (databases, sensors, surveys, files, etc.) is often **incomplete** (missing values), **inaccurate** (errors), **inconsistent** (different formats or units), or **noisy** (random variations or outliers). If such data is used directly, it can lead to **misleading analysis results** or **failure of mining algorithms**.

Data preprocessing ensures that:

- The data is **clean** (free from errors and noise).
- The data is **integrated** (combined from multiple sources consistently).
- The data is **transformed** (normalized, scaled, or aggregated for better mining performance).
- The data is **reduced** (removing redundant attributes or compressing without losing essential patterns).

DATA CLEANING:

Data cleaning is the process of identifying and correcting (or removing) errors, inconsistencies, and missing data to improve the overall quality of a dataset before analysis.

It ensures that the data is **accurate, complete, consistent, and reliable** so that mining algorithms can produce meaningful results.

Importance

- **Accuracy** → Incorrect data leads to wrong conclusions.
- **Completeness** → Missing values reduce the usefulness of data.
- **Consistency** → Avoids conflicts between datasets.
- **Efficiency** → Clean data reduces processing time.

Data cleaning has 2 techniques

1) handling missing values

HANDLING MISSING VALUES:

In this method there are five common data cleaning methods

1. remove tuple
2. fill manually

3. fill mean value
4. fill mean value to all
5. fill constant value

Let us take an example data set to illustrate each method:

ID	Name	Age	Salary
1	Ravi	25	50000
2	Priya	NULL	60000
3	Amit	28	45000
4	Hari	35	47000
5	Sneha	NULL	NULL
6	Rajesh	30	52000

1. Remove Tuples (Record Deletion)

If any row has missing data, delete the whole row. Works when missing data is rare

This method deletes records containing missing values. It's useful when the dataset is large and missing values are very few, so deleting them won't harm analysis.

It avoids adding fake data, but if many rows have missing values, this method can remove a lot of useful information.

Here, rows with missing Age or Salary are deleted.

From our dataset: Row 2 (Age = NULL) and Row 5 (Age & Salary = NULL) are removed.

Result:

ID	Name	Age	Salary
1	Ravi	25	50000
3	Amit	28	45000
4	Hari	35	47000
6	Rajesh	30	52000

2. Fill Manually

Check original sources (forms, files, contacts) and fill the missing values yourself.

This approach gets the real missing value from an official source, making it the most accurate method.

It is time-consuming and impractical for large datasets, but works well for small and important datasets.

Example: If a student's date of birth is missing, the school checks admission records to get it

- priya's Age = 27
- Sneha's Age = 26, Salary = ₹55,000 We fill these in.

Result:

ID Name Age Salary

1	Ravi	25	50000
2	Priya	27	60000
3	Amit	28	45000
4	Hari	35	47000
5	Sneha	26	55000
6	Rajesh	30	52000

3. Fill Mean Value

Replace the missing number with the average of the other numbers in that column.
Here, we calculate the mean (average) of available values and replace the missing entry with it. This is fast, keeps dataset size the same, and works for numeric data without extreme outliers.

Replace missing number with the average of the existing numbers in that column. For **Age**: Available = 25, 28, 35, 30 → Mean = (25+28+35+30)/4 = 29.5 Replace missing
Ages (Row 2 & 5) with 29.5.

Result:

ID	Name	Age	Salary
1	Ravi	25	50000
2	Priya	29.5	60000
3	Amit	28	45000
4	Hari	35	47000
5	Sneha	29.5	NULL
6	Rajesh	30	52000

4. Fill Mean Value to All (Global Mean Filling)

Find one single average for the whole dataset and fill all missing values with it.
Unlike normal mean filling (which might be done per group), this method uses one global mean for all missing entries. It's simple but may ignore group differences.

find one overall mean and use it for all missing values.

For **Salary**: Available = 50000, 60000, 45000, 47000, 52000 Mean = (50000 + 60000 + 45000 + 47000 + 52000) / 5 = 50800 Replace missing Salary (Row 5) with 50800.

Result:

ID	Name	Age	Salary
1	Ravi	25	50000
2	Priya	29.5	60000
3	Amit	28	45000
4	Hari	35	47000
5	Sneha	29.5	50800
6	Rajesh	30	52000

5. Fill Constant Value

Put a fixed value (like 0 or "Unknown") in place of missing data. Replace missing values with a constant decided by the analyst.

Good for categorical data or when we want to clearly mark missing values.

Use a fixed value (like 0 or "Unknown") for all missing entries. Let's set: Age = 0, Salary = 0 for missing values.

Result:

ID	Name	Age	Salary
1	Ravi	25	50000
2	Priya	0	60000
3	Amit	28	45000
4	Hari	35	47000
5	Sneha	0	0
6	Rajesh	30	52000

NOISY DATA:

1. **What is Noisy Data?**

Definition:

Noisy data is data that contains random errors, outliers, or meaningless variations that do not represent the true values of the attributes. Noise hides patterns, reduces accuracy, and can mislead

analysis if not handled.

Example:

In our dataset:

ID Name Age Salary

2 Priya 27 60000

3 Amit 28 45000

4 Hari 35 47000

1 Ravi 25 50000

5 Sneha 26 55000

6 Rajesh 60 52000

Here, **Age = 60** may be unrealistic for a dataset of young employees and could be considered noise.

Causes of Noisy Data

- Human errors during data entry
- Faulty measuring instruments
- Transmission errors in sensors or networks
- Outdated or inconsistent records
- Random fluctuations in data collection

Types of Noisy Data Handling Techniques

A) Binning

Binning is a data smoothing technique used in data preprocessing to reduce the effects of minor observation errors or noise In the dataset.

Smoothing by grouping data into bins, then replacing each value with a representative value. (Ages sorted: 25, 26, 27 , 28, 35, 60)

Bin1(25,26,27) Bin 2(28,35,60)

1. **Mean Binning** – Replace values with bin mean.

This technique reduces within the bin and smooths noisy data

Example (Ages sorted: 25, 26, 27 , 28, 35, 60; bin size=3):

- Bin 1 mean = 26 → [26, 26, 26]
- Bin 2 mean = 41 → [41, 41, 41]

2. **Median Binning** – Replace each value in a bin with bin median.Median is the middle value when numbers are arranged in order .it is less sensitive to outliers compared to mean

Median preserves the central location without being affected by large deviations

- Bin 1 median = 26, Bin 2 median = 35 → [26, 26, 26, 35, 35, 35]

3. **Boundary Binning –** Replace with nearest boundary.

o **Bin 1 boundaries = 25, 27; Bin 2 boundaries = 28, 60 B)REGRESSION:**

Regression is a **statistical method** that estimates the relationship between a dependent variable (target) and one or more independent variables (predictors).

When used for noisy data, regression predicts the "expected" value for a given record, replacing or smoothing out noisy entries.

- Fits a mathematical function (line or curve) to the dataset.
- Identifies deviations (noise) from this trend.
- Adjusts or replaces noisy values with predicted ones. Regression is of two types i.linear regression ii.multiple regression **1. Linear Regression**

A statistical technique to predict the dependent variable using a single independent variable by fitting a straight line. Formula:

$Y=mX+cY = mX + cY=mX+c$

- Used to identify trends and replace noisy data with predicted values.
- Example: Predicting weight from height, replacing outlier weights with model-predicted values.

2. Multiple Linear Regression

An extension of linear regression that uses two or more independent variables to predict the dependent variable. Formula:

$Y=aX1+bX2+cY = aX\_1 + bX\_2 + cY=aX1+bX2+c$

- More accurate as it considers multiple influencing factors.
- Example: Predicting house price using area and number of bedrooms, replacing unrealistic values.

Data Integration

Definition:

Data Integration combines data from multiple sources into a single, unified dataset for analysis. It resolves differences in schema, removes redundancy, and ensures consistency.

a) Schema Integration

- Merging metadata from different sources into a common schema.
- **Types of Conflicts:**

1. **Naming Conflicts** – Different names for the same attribute (e.g., "Cust_ID" vs

"CustomerID").

2. **Data Type Conflicts –** Same attribute stored in different formats (e.g., int vs string).
3. **Structural Conflicts –** Same data stored in different structures.

b) Entity Identification Problem

- Determining if data from different sources refer to the same real-world entity.

- **Techniques:**
    1. String matching (e.g., "R. Kumar" = "Rajesh Kumar").
    2. Using unique identifiers (e.g., Aadhaar ID, Employee ID).

c) Redundancy Detection

- Finding and removing duplicate data.
- **Types:**
    1. **Exact duplicates** – Same record repeated.
    2. **Partial duplicates** – Slight differences but same meaning.

d) Data Value Conflicts

- Differences in the same attribute's value across sources.
- **Types:**
    1. **Format Conflicts** – Date formats DD/MM/YYYY vs MM-DD-YYYY.
    2. **Measurement Conflicts** – Weight in kg vs lbs.

2. Data Transformation

Definition:

Data Transformation is the process of converting data into a suitable format for analysis. It improves data quality, consistency, and compatibility for mining algorithms.

Common operations include **smoothing**, **aggregation**, **generalization**, **normalization**, and **attribute construction**.

It helps remove noise, scale values, and derive new meaningful features. **a) Smoothing**

- Removes noise from data.
- **Types:**
    1. **Binning** – Mean, Median, Boundaries.
    2. **Regression** – Linear, Multiple.
    3. **Clustering** – Grouping similar values.

b) Aggregation

- Summarizing or combining data.
- **Types:**

2. **Temporal Aggregation –** Combining data by time period.

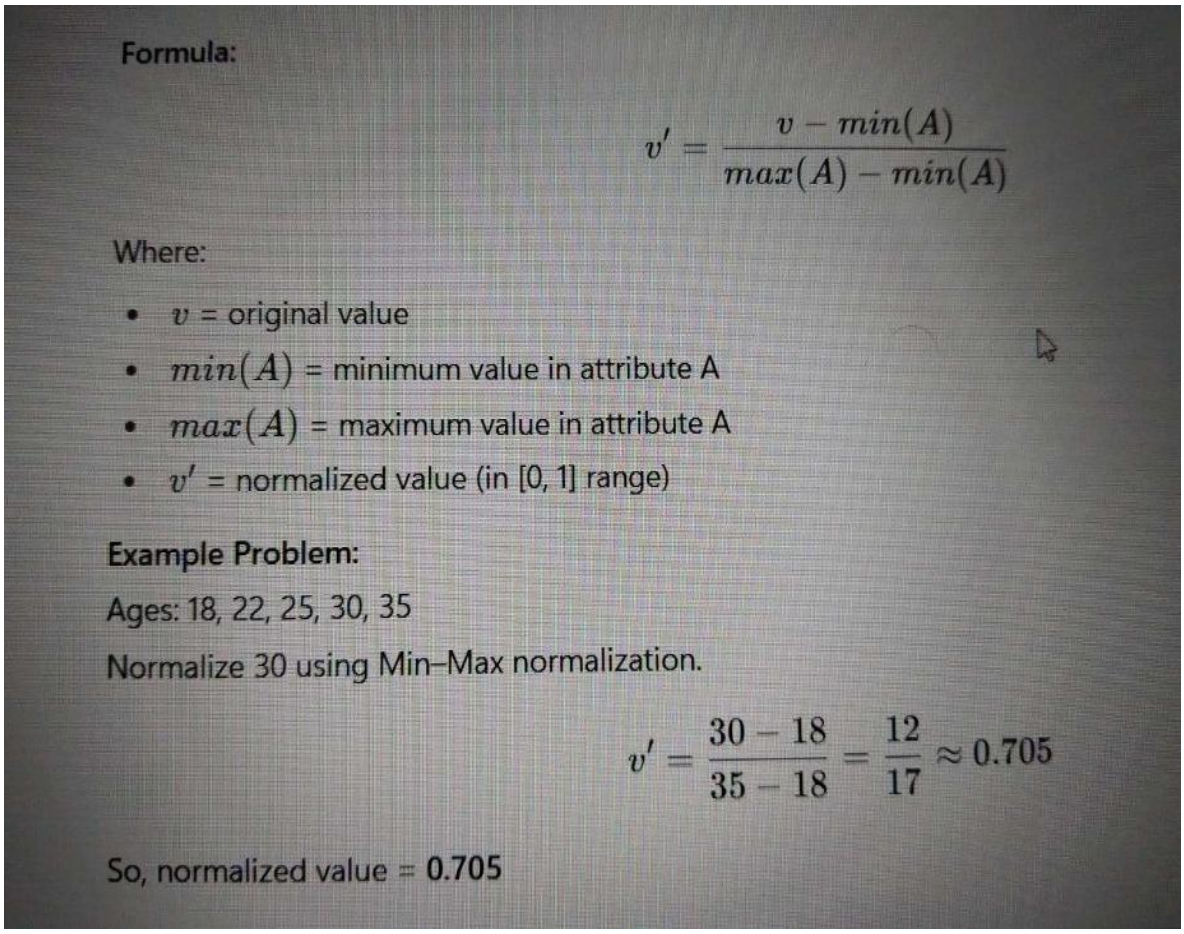
c) **Normalization**

- **Data Normalization** is the process of scaling numerical data into a standard range like **[0, 1]** or with mean 0 and standard deviation 1.

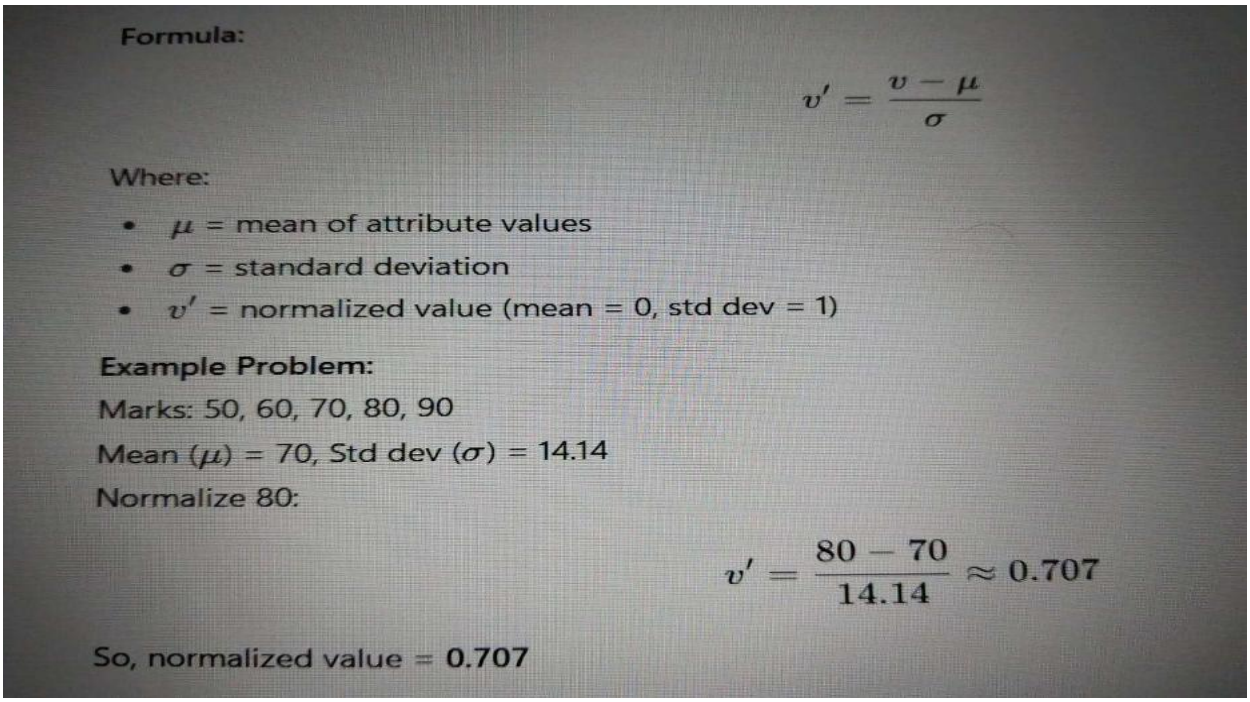
It ensures that all attributes contribute equally during analysis and prevents dominance by large-value features..

Types:

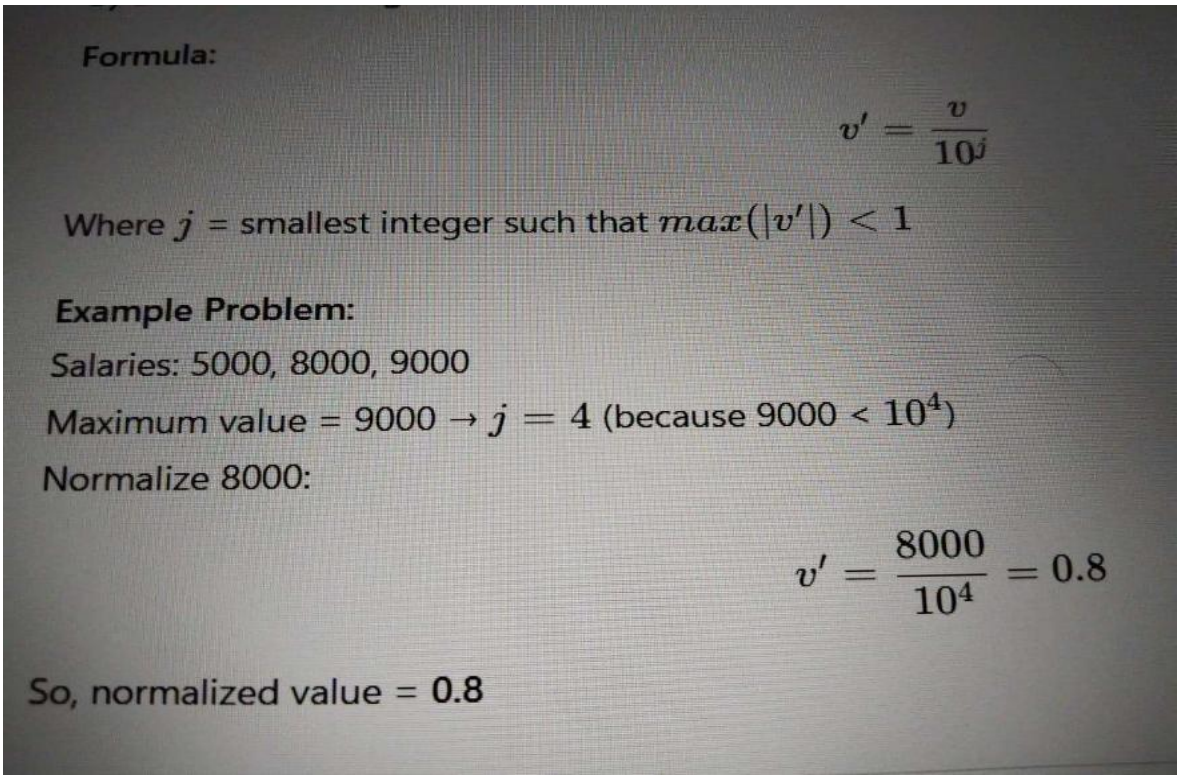
Min-Max Normalization – Scales values to a fixed range (usually [0, 1]) using the minimum and maximum of the attribute



Z-Score Normalization –



Decimal Scaling – Moves the decimal point of values by a factor of 10 until all values fall in



the range (-1, 1).

e) Attribute Construction (Feature

- Creating new attributes from existing ones.
- **Example:** From "Date of Birth" → create "Age" attribute.

Data Reduction:

Definition:

Data Reduction is the process of reducing the volume of data while maintaining its integrity and analytical value.

It makes data mining faster, saves storage, and reduces processing costs.

Main Goal:

- Remove irrelevant or redundant data.
- Keep only the most important information for analysis.

1. Dimensionality Reduction

- Reduces the number of attributes (features) in the dataset.
- Removes irrelevant, redundant, or highly correlated features.

- **Techniques:**
    1. **Feature Selection** – Select the most relevant attributes.
    2. **Feature Extraction** – Create new features from existing ones (e.g., PCA).
- **Example:** From 100 survey questions, keeping only 20 most useful for prediction.

2. Numerosity Reduction

- Reduces data volume by replacing detailed data with smaller, more descriptive representations.
- **Techniques:**
    1. **Parametric Methods** – Use models like regression to represent data.
    2. **Non-Parametric Methods** – Use histograms, clustering, sampling.
- **Example:** Instead of storing all sales data, store only sales trends or summaries.

3. Data Compression

- Encodes data into fewer bits while preserving information.
- **Techniques:** Lossless (no data loss) & Lossy (some loss but acceptable).
- **Example:** Compressing image dataset to speed up image mining.

Feature Selection

Definition:

Feature Selection is the process of choosing the most relevant features from a dataset for analysis or modeling.

It helps improve model accuracy, reduce overfitting, and speed up computation.

Types of Feature Selection Methods:

1. **Filter Methods** – Select features based on statistical measures (e.g., correlation, Chi-square test).
2. **Wrapper Methods** – Select features by testing their impact on model performance (e.g., forward selection, backward elimination).
3. **Embedded Methods** – Feature selection is built into the learning algorithm (e.g., LASSO regression, decision trees).

Example:

If predicting student grades, we may remove "favorite color" and keep "attendance" and "study hours" as features.

Data Discretization :

Definition:

Data Discretization is the process of converting continuous numerical data into discrete intervals or categories.

It helps reduce data size, simplify patterns, and improve algorithm performance (especially in classification).

Why Needed:

- Makes continuous data easier to interpret.

- Reduces computational complexity.
- Required by algorithms that work only on categorical data.

Types of Data Discretization:

a) Binning

- Groups continuous values into bins (intervals).

Types:

1. **Equal-Width Binning** – All bins have the same range.
    - Example: Scores 0–100 in bins of width 20 → (0–20), (21–40), etc.
2. **Equal-Frequency Binning** – Each bin has the same number of data points.
    - Example: 10 students' marks → 5 in bin 1, 5 in bin 2.
3. **Custom Binning** – Based on domain knowledge.
    - Example: Age groups → Child (0–12), Teen (13–19), Adult (20+).

b) Histogram Analysis

- Divides data based on histogram bars.
- Bins are created according to frequency distribution.

c) Clustering-Based Discretization

- Uses clustering (like K-means) to group similar values, then assigns them discrete labels.

d) Decision Tree Analysis

- Splits data based on decision tree thresholds.

2. Data Normalization

Definition:

Data Normalization is scaling numerical data into a standard range or format so all attributes contribute equally to analysis.

Types:

a) **Min–Max Normalization**

b) **Z–Score Normalization**

c) **Decimal Scaling**

Measures of Similarity and Dissimilarity are statistical or mathematical tools used to **quantify how alike or different** two objects, data points, or datasets are. They are widely used in **machine learning, data mining, clustering, pattern recognition, and information retrieval**.

1. **Basic Concepts**

- **Similarity** o Measures how close or alike two objects are. o Higher value → more similar.
    - o Range often **0 to 1** (1 = identical, 0 = completely different).
- **Dissimilarity (Distance)** o Measures how far apart two objects are. o Lower value → more similar.
    - o Usually **≥ 0** (0 = identical, larger values = more different).

Relationship:

Similarity=1−Normalized Dissimilarity\text{Similarity} = 1 - \text{Normalized Dissimilarity} Dissimilarity=1−Similarity\text{Dissimilarity} = 1 - \text{Similarity}

(depending on normalization)

2. **Common Measures**

A. Similarity Measures

1. **Cosine Similarity** (for text/data vectors)

Cosine Similarity=A⋅B∥A∥∥B∥\text{Cosine Similarity} = \frac{A \cdot B}{\|A\| \|B\|}

o Measures angle between two vectors (0° = identical).

- Used in **text mining, recommender systems**.

2. **Jaccard Similarity** (for sets)

$J(A,B)=|A∩B||A∪B|J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ o Measures overlap between sets.

- Used in **categorical data, market basket analysis**.

3. **Pearson Correlation Coefficient** $r=∑(xi−x¯)(yi−y¯)∑(xi−x¯)2∑(yi−y¯)2r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$

- Measures linear relationship (-1 to 1).
- Used in **statistics, regression analysis**.

4. **Dice Similarity Coefficient (DSC)**

$DSC=2|A∩B||A|+|B|DSC = \frac{2|A \cap B|}{|A| + |B|}$

o Similar to Jaccard but gives more weight to common elements.

B. Dissimilarity / Distance Measures

1. **Euclidean Distance**

$d=∑i=1n(xi−yi)2d = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$ o Straight-line distance in nn-dimensional space.

2. **Manhattan Distance** (L1 Norm) d=∑i=1n|xi−yi|d = \sum_{i=1}^n |x_i - y_i| o Distance measured along axes (grid-like path).

3. **Minkowski Distance** (Generalized) d=(∑i=1n|xi−yi|p)1/pd = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p} o

p=1

p=1 → Manhattan, p=2p=2 → Euclidean.

4. **Hamming Distance**

o Number of positions at which corresponding symbols differ.

- Used for **binary/categorical data, error detection**.

5. **Mahalanobis Distance**

d=(x−μ)TS−1(x−μ)d = \sqrt{(x - \mu)^T S^{-1} (x - \mu)} o Accounts for correlation between variables.

- Used in **multivariate anomaly detection**.

3. Application Areas

- **Clustering** → K-means uses Euclidean, hierarchical clustering uses various measures.
- **Recommendation Systems** → Cosine similarity, Pearson correlation.
- **Image Processing** → Structural similarity index (SSIM).
- **Text Mining** → Cosine, Jaccard.

- **Bioinformatics** → Sequence alignment similarity measures.

4. **Comparison Table**

Measure Type	Example Measures	Data Type	Range
Similarity	Cosine, Jaccard, Pearson	Vector, Set, Mixed	0 to 1 (or -1 to 1 for Pearson)
Dissimilarity	Euclidean, Manhattan, Hamming	Numeric, Binary	≥ 0

1. What is Exploratory Data Analysis (EDA)? Definition:

Exploratory Data Analysis is the **process of examining and understanding data** before formal modeling. It helps you:

- Understand **patterns, trends, and relationships**.
- Detect **missing values, outliers, anomalies**.
- Decide which statistical models or preprocessing steps to apply.

Origin:

Coined by **John Tukey** (1977) as a philosophy of letting the data speak for itself before making assumptions.

2. Steps in EDA

1. **Data Collection**

o Importing datasets (CSV, Excel, database, API). o Example: pd.read_csv('data.csv')

2. **Data Cleaning** o Handle missing values (NaN), duplicates, incorrect types. o Example: .dropna(), .fillna()

3. **Data Profiling o Overview of structure & summary statistics:**

- df.info()
- df.describe()

4. **Univariate Analysis** *(single variable)* o Check distribution, central tendency, spread.

o Tools: Histograms, boxplots.

5. **Bivariate/Multivariate Analysis** *(relationship between variables)* o Correlation matrix, scatterplots, heatmaps.

6. **Outlier Detection** o IQR method, Z-score, visualization via boxplots.

7. **Feature Engineering** o Creating new features or transforming existing ones.

3. Data Visualization in EDA

Visualization is the **graphical representation of data** to see patterns more clearly.

A. Types of Plots

Type	Purpose	Example Tools
Histogram	Distribution of one variable	Matplotlib, Seaborn
Boxplot	Spread, outliers	Seaborn
Bar Chart	Categorical data counts	Matplotlib
Scatter Plot	Relationship between two variables	Matplotlib, Plotly
Heatmap	Correlation matrix visualization	Seaborn
Pie Chart	Percentage breakdown	Matplotlib
Pairplot	Multiple variable relationships	Seaborn

B. Example in Python

```
import pandas as pd
import seaborn as sns import matplotlib.pyplot as plt
```

```
# Load dataset df =
pd.read_csv("data.cs
v")

# Summary
print(df.info())
print(df.describe(
))

# Histogram
sns.histplot(df['age'],
kde=True) plt.show()

# Boxplot
sns.boxplot(x=df['sa
lary']) plt.show()

# Scatterplot
sns.scatterplot(x='age', y='salary',
data=df) plt.show()
```

```
# Heatmap of correlations
sns.heatmap(df.corr(), annot=True,
cmap="coolwarm") plt.show()
```

4. **Importance of EDA & Visualization**
   - **Identifies trends & patterns** → better decision-making.
   - **Reveals data quality issues** → ensures model reliability.
   - **Supports feature selection** → reduces noise.
   - **Improves communication** → visual reports are easier to understand.
5. **EDA vs Statistical Modeling**

EDA	**Modeling**
No strict assumptions	Requires assumptions
Focus on discovery	Focus on prediction
Graphs & summaries	Equations & parameters

UNIT-III: Association Rule Mining and Classification

1. Basics of Association Rule Mining

Association Rule Mining (ARM) is a data mining method used to find hidden patterns, correlations, and relationships in large datasets.

It is most applied in market basket analysis – e.g., supermarkets analyze transaction data to see which products are often purchased together.

1.1 Key Measures in Association Rules

To evaluate the strength of association rules, three main metrics are used: Support, Confidence, and Lift.

(a) Support (s)

- Measures how often an itemset appears in the database.
- **Definition:**

$$Support(X \Rightarrow Y) = \frac{\text{Number of transactions containing } (X \cup Y)}{\text{Total number of transactions}}$$

- **Interpretation:**

Higher support = the itemset occurs frequently in the dataset.

(b) Confidence (c)

- Measures how often items in Y appear in transactions that contain X.
- **Definition:**

$$Confidence(X \Rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)}$$

- **Interpretation:**

Confidence represents the probability of purchasing Y when X is purchased.

(c) Lift (L)

- Compares the observed probability of X and Y appearing together with what would be expected if they were independent.
- **Definition:**

$$Lift(X \Rightarrow Y) = \frac{Confidence(X \Rightarrow Y)}{Support(Y)}$$

- **Interpretation:**
  - Lift > 1 → Positive correlation (X and Y occur together more than expected).
  - Lift = 1 → X and Y are independent.

- Lift < 1 → Negative correlation (X and Y occur together less than expected).

Example:

Supermarket transactions data:

- Total transactions = 1000
- 200 contain {Bread, Milk}
- 250 contain Bread
- 400 contain Milk

Step 1: Support

$$Support(Bread \Rightarrow Milk) = \frac{200}{1000} = 0.2 \quad (20\%)$$

Step 2: Confidence

$$Confidence(Bread \Rightarrow Milk) = \frac{200}{250} = 0.8 \quad (80\%)$$

Step 3: Lift

$$Lift = \frac{0.8}{0.4} = 2.0$$

Interpretation:

- The rule Bread ⇒ Milk is strong.
- Lift = 2 means customers who buy bread are twice as likely to buy milk compared to random chance.

1. Apriori Algorithm

Apriori is a classic algorithm for frequent itemset mining and association rule learning.
It is widely used in Market Basket Analysis.

1.1 Principle

- **Based on the Apriori Property:**

"All non-empty subsets of a frequent itemset must also be frequent."

- Meaning: If {Milk, Bread} is frequent, then {Milk} and {Bread} must also be frequent.

1.2 Steps of Apriori Algorithm

1. **Find Frequent 1-Itemsets (L1):**
   - Scan the database to find items with support ≥ minsup (minimum support).
2. **Generate Candidate Itemsets (Ck):**
   - Use frequent itemsets of size *(k-1)* to generate candidates of size *k*.
3. **Prune Step:**
   - Remove candidate itemsets whose subsets are not frequent.
4. **Count Support**

- Scan database again to count support for the remaining candidates.

5. Repeat until no more frequent itemsets can be generated.
6. **Generate Association Rules:**
   - From frequent itemsets, generate rules using Confidence and Lift.

1.3 Example of Apriori

Transactions:

T1 = {Bread, Milk, Butter}

T2 = {Milk,

Bread} T3 =

{Milk, Butter}

T4 = {Bread,

Butter}

T5 = {Milk, Bread, Butter}

Step 1 – Frequent 1-Itemsets (L1):

- Bread = 4/5 = 0.8
- Milk = 4/5 = 0.8
- Butter = 4/5 = 0.8

(All ≥ 0.5 minsup → keep

them) Step 2 – Candidate 2-

Itemsets (C2):

- {Bread, Milk} = 3/5 = 0.6
- {Bread, Butter} = 3/5 = 0.6
- **{Milk, Butter} =**

3/5 = 0.6 Step 3 –

Candidate 3-Itemset

(C3):

- {Bread, Milk, Butter} = 2/5 = 0.4 (<0.5 →

eliminate) Step 4 – Association Rules:

- Bread ⇒ Milk (Confidence = 3/4 = 0.75)
- Milk ⇒ Bread (Confidence = 3/4 = 0.75)
- Bread ⇒ Butter (Confidence = 3/4 = 0.75)

1.4 Advantages &

Disadvantages Advantages:

- Simple and easy to implement.
- Produces all frequent itemsets.
- Widely applicable (market basket, fraud detection, bioinformatics).

Disadvantages:

- Requires multiple database scans (slow for large data).
- Generates too many candidate sets.
- Memory and time-intensive.

2. FP-Growth Algorithm

FP-Growth (Frequent Pattern Growth) is an improved algorithm for mining frequent

itemsets. Unlike Apriori, FP-Growth does not generate candidate sets → it compresses the dataset into an FP-Tree (Frequent Pattern Tree).

2.1 Principle

- Uses a divide-and-conquer approach.
- Database is compressed into an FP-Tree.
- Frequent itemsets are mined recursively by exploring conditional pattern bases.

2.2 Steps of FP-Growth Algorithm

1. **Scan Database & Find Frequent Items:**
    - Count item frequencies.
    - Discard infrequent items (support < minsup).
2. **Build FP-Tree:**
    - Order frequent items by descending support.
    - Insert transactions into the FP-Tree.
    - Each path in the tree represents a transaction.
3. **Mine FP-Tree:**

- For each item, extract its conditional pattern base (set of prefix paths).
- Build conditional FP-Tree from the pattern base.
- Recursively mine frequent patterns.

2.3 Example of FP-Growth

Transactions:

T1 = {Milk, Bread,

Butter} T2 = {Milk,

Bread}

T3 = {Milk,

Butter} T4 =

{Bread, Butter}

T5 = {Milk, Bread,

Butter} Step 1 –

Find Frequent

Items:

- Milk = 4, Bread = 4, Butter = 4

Step 2 – Build FP-Tree (descending frequency order: Milk → Bread → Butter):

- Root → Milk → Bread → Butter
- Compact tree structure built for all

transactions. Step 3 – Mine FP-Tree:

- Generate frequent patterns directly from tree paths.
- Example: {Milk, Bread}, {Milk, Butter}, {Bread, Butter}, {Milk, Bread, Butter}.

2.4 Advantages &

Disadvantages Advantages:

- Only 2 database scans required.
- No candidate generation → more efficient.
- Works well for large datasets.

Disadvantages:

- FP-Tree may be very large if dataset is huge.
- More complex implementation than Apriori.

3. Comparison: Apriori vs FP-Growth

Feature	Apriori	FP-Growth
Candidate Generation	Yes	No
Database Scans	Many	2
Memory Usage	Low	Higher

Speed	Slower	Faster
Implementation Simplicity	Easy	Complex

1.4 Applications of Association Rule Mining

- **Market Basket Analysis** – Products bought together.
- **Cross-Selling & Recommendation Systems** – Amazon, Flipkart, Netflix.
- **Fraud Detection** – Suspicious patterns in transactions.
- **Medical Diagnosis** – Symptoms and diseases association.
- **Web Usage Mining** – User navigation patterns.

2. Classification Techniques

Classification is a **supervised learning technique** used to assign items into predefined classes (labels) based on their attributes.

Decision Trees

A Decision Tree is a supervised learning technique used for classification and sometimes regression.

- **Each internal node: tests an attribute.**
- **Each branch: outcome of the test.**
- **Each leaf node: class label.**

It works like a flowchart to classify data step by step.

1. ID3 Algorithm (Iterative

Dichotomiser 3) Developed by

Ross Quinlan (1986).

It builds a decision tree using the top-down greedy search approach.

1.1 Principle

- **Uses Information Gain based on Entropy as a measure to select attributes.**
- **Attribute with highest Information Gain is chosen for splitting.**

1.2 Important Formulas

Entropy (measure of impurity):

$$Entropy(S) = -\sum_{i=1}^{n} p_i \cdot \log_2(p_i)$$

where p_i = probability of class i.

Information Gain (IG):

$$IG(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot Entropy(S_v)$$

where S_v is subset of S for which attribute A = value v.

1.3 Steps of ID3

1. Calculate Entropy of dataset.
2. For each attribute, calculate Information Gain.
3. Select attribute with highest Information Gain as root node.
4. Split dataset according to the chosen attribute.
5. **Repeat recursively until:**
   - All samples are classified OR
   - No attributes remain.

1.4 Example of ID3

Dataset: Weather (Play Tennis)

Outlook	Temperature	Humidity	Windy	Play

Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rain	Mild	High	False	Yes
Rain	Cool	Normal	False	Yes

- Step 1: Compute Entropy(Play).
- Step 2: Compute IG for Outlook, Temperature, Humidity, Windy.

- Step 3: Choose Outlook as root (highest IG).
- Step 4: Repeat for subsets.

tree is formed with Outlook at root, branches leading to Play = Yes/No

1.5 Advantages of ID3

Simple to understand and implement. Works well for categorical data.

1.6Disadvantages of ID3

Prone to overfitting. Cannot handle continuous attributes directly. Sensitive to noisy data.

2. C4.5 Algorithm

C4.5 is an extension of ID3, also developed by Ross Quinlan (1993). It improves ID3 by addressing its limitations.

2.1 Improvements over ID3

- Handles continuous attributes (by creating thresholds).
- Handles missing values.
- Uses Gain Ratio instead of pure Information Gain (avoids bias towards attributes with many values).
- Includes Pruning to avoid overfitting.

2.2 Important Formulas

Split Information (SI):

$$SI(A) = -\sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot \log_2\left(\frac{|S_v|}{|S|}\right)$$

Gain Ratio (GR):

$$GR(S, A) - \frac{IG(S, A)}{SI(A)}$$

👉 Attribute with highest **Gain Ratio** is chosen.

2.3 Steps of C4.5

1. Compute Entropy of dataset.
2. For each attribute, compute Information Gain and Split Information.
3. Calculate Gain Ratio.
4. Choose attribute with highest Gain Ratio.
5. Handle continuous attributes by selecting split points (e.g., Temperature ≤ 75).
6. Handle missing values using probability distribution.
7. Apply pruning to simplify tree and reduce overfitting.

Handles both categorical & continuous data. Handles missing values.
Produces smaller, accurate trees (due to pruning). More robust than ID3.

2.5 Disadvantages of C4.5

Computationally expensive for large datasets. May create biased trees if data is imbalanced.

3. ID3 vs C4.5

Feature	ID3	C4.5
Attribute Type	Categorical only	Both categorical & continuous

Feature	ID3	C4.5
Split Criterion	Information Gain	Gain Ratio
Missing Values	Not handled	Handled
Pruning	No	Yes
Overfitting	More prone	Less prone
Efficiency	**Faster**	**Slower**

2.2 Bayesian Classifiers Based on Bayes' Theorem:

$$P(H|X) = \frac{P(X|H) \cdot P(H)}{P(X)}$$

Where:

- (P(H|X)): Posterior probability of hypothesis H given data X.
- (P(X|H)): Likelihood of data X given hypothesis H.
- (P(H)): Prior probability of hypothesis.
- (P(X)): Probability of data.

2.3 Naïve Bayes Classifier

- Assumes **independence** among features.
- Despite its simplicity, performs well in practice.

Steps:

1. Calculate prior probability for each class.
2. Compute conditional probability for each attribute given the class.
3. Apply Bayes' theorem to classify new data.

Applications:

- Spam filtering
- Sentiment analysis
- Document classification

Advantages: Simple, works well with high-dimensional data.

Disadvantages: Assumption of independence may not hold.

2.4 Rule-Based Classification

- Classification done using a set of **IF–THEN rules**.
- **Example:**
  - IF outlook = sunny AND humidity = high THEN play = no
  - IF outlook = rainy AND windy = false THEN play = yes

Advantages: Interpretable, easy to implement.

Disadvantages: Large number of rules may become unmanageable.

2.5 Model-Based Classification

- Instead of rules, build a **mathematical/statistical model** for classification.
- Examples: Decision Trees, Bayesian models, SVMs, Neural Networks.

Advantages: Generalizes better, compact representation.

Disadvantages: Harder to interpret compared to rule-based.

UNIT-IV: Clustering and Advanced Mining Methods

1. Introduction to Clustering

Clustering is an unsupervised learning technique used in data mining where a set of objects is grouped into clusters in such a way that:

- Objects in the same cluster are highly similar (high intra-cluster similarity).
- Objects in different clusters are dissimilar (low inter-cluster similarity).

Clustering does not rely on predefined labels or classes. Instead, it discovers hidden patterns and natural groupings in data.

1.1 Types of Clustering

1. Partitioning Clustering
   - Divides data into *k* non-overlapping clusters.
   - Each object belongs to exactly one cluster.
   - Algorithms try to minimize an objective function, such as the sum of squared errors (SSE).
   - **Examples:**
     - *K-Means:* uses centroids (mean values) to represent clusters.
     - *K-Medoids:* uses actual data points (medoids) as cluster centers, more robust to outliers.
2. Hierarchical Clustering
   - Produces a tree-like structure (dendrogram) showing nested clusters.
   - **Two approaches:**
     - Agglomerative (bottom-up): Start with each object as a single cluster, then merge them step by step.
     - Divisive (top-down): Start with one large cluster and split into smaller clusters.
   - Linkage criteria (e.g., single linkage, complete linkage, Ward's method) decide how to merge clusters.
3. Density-Based Clustering
   - Clusters are formed around dense regions of points.
   - Can detect arbitrary-shaped clusters and handle noise/outliers.
   - **Examples:**
     - ***DBSCAN (Density-Based Spatial Clustering of Applications with Noise):***
       forms clusters from core points and density reachability.
     - *OPTICS (Ordering Points To Identify the Clustering*

Structure): handles clusters of varying density better than DBSCAN.

4. Grid-Based Clustering
    - The data space is divided into a finite number of grid cells.
    - Clustering is performed on grids instead of individual points, making it efficient for large datasets.
    - **Examples:**
        - *STING (Statistical Information Grid):* uses hierarchical grids with statistical information.
        - *CLIQUE:* combines density and grid approaches, effective for high-dimensional data.

1.2 Applications of Clustering

1. Market Segmentation
    - Grouping customers based on purchasing behavior.
    - Helps in personalized marketing, product recommendations, and customer profiling.

2. Document Clustering
    - Groups similar articles, research papers, or web pages.
    - Useful in search engines, recommendation systems, and topic modeling.
3. Image Segmentation
    - Divides an image into meaningful regions (e.g., separating background from objects).
    - Helps in medical imaging, object recognition, and computer vision tasks.
4. Medical Data Analysis
    - Groups patients based on symptoms, genetic information, or medical history.
    - Useful for disease diagnosis, drug discovery, and treatment planning.
5. Anomaly Detection
    - Identifies unusual patterns or outliers that do not belong to any cluster.
    - Applied in fraud detection, intrusion detection, and fault detection in systems.

✅ **Summary:**

Clustering is a powerful tool in data mining that helps uncover hidden structures in data. Various methods (partitioning, hierarchical, density-based, and grid-based) suit different types of datasets and requirements. Its applications span across business, healthcare, computer vision, and security domains.

2. Partitioning Methods

Partitioning methods divide a dataset into k clusters, where each data point belongs to the cluster with the most similarity. They aim to minimize intra-cluster variation and maximize inter-cluster separation.

2.1 K-Means Clustering

- A widely used clustering algorithm that minimizes within-cluster variance. Algorithm Steps:

1. Choose k initial centroids (randomly or using heuristics like k-means++).
2. Assign each data point to the nearest centroid (based on distance, usually Euclidean).
3. Recalculate the centroid of each cluster as the mean of its points.
4. Repeat steps 2–3 until centroids no longer change significantly (convergence). Advantages:

- Simple to implement and understand.
- Efficient for large datasets.
- Works well when clusters are compact and well-separated. Disadvantages:

- Sensitive to initial centroid selection (may lead to local optima).
- Works best for spherical or convex clusters.
- Requires specifying k in advance.
- Poor performance with outliers and non-linear cluster shapes.

2.2 K-Medoids (PAM – Partitioning Around Medoids)

- A variant of K-Means where clusters are represented by medoids (actual data points) instead of centroids (average positions).
- Reduces sensitivity to outliers since medoids are real objects. Algorithm Steps:

1. Select k random medoids from the dataset.
2. Assign each data point to the nearest medoid.
3. Swap a medoid with a non-medoid point if it reduces the total clustering cost (sum of dissimilarities).
4. **Repeat until medoids remain stable. Advantages:**

- More robust to noise and outliers compared to K-Means.
- Works with arbitrary distance measures (not limited to Euclidean). Disadvantages:
  - More computationally expensive than K-Means (especially for large datasets).
  - Not as scalable for high-dimensional or very large data.

3. Hierarchical Clustering

Hierarchical clustering builds a tree-like structure (dendrogram) to represent nested clusters in the dataset. Unlike partitioning methods, it does not require the number of clusters in advance.

3.1 Agglomerative (Bottom-Up)

- Starts with each data point as a separate cluster.
- Iteratively merges the closest pair of clusters until all points are combined into a single cluster or a stopping criterion is met.

Linkage Criteria (to define cluster distance):

1. **Single Linkage (Nearest Neighbor):**
   - Distance between two clusters = minimum distance between any two points in the clusters.
2. **Complete Linkage (Farthest Neighbor):**
   - Distance between two clusters = maximum distance between any two points.
3. **Average Linkage (UPGMA):**
   - Distance between clusters = average distance between all pairs of points.
4. **Ward's Method:**
   - Minimizes the increase in total within-cluster variance after merging. Advantages:
     - Produces a dendrogram, which is useful for visualizing cluster hierarchy.
     - **Does not require the number of clusters beforehand. Disadvantages:**
       - High computational cost ($O(n^2)$ memory and time for naive implementation).
       - Sensitive to noise and outliers.

3.2 Divisive (Top-Down)

- Starts with one single cluster containing all points.

- Iteratively splits clusters into smaller clusters based on dissimilarity until each cluster contains a single point or stopping criteria are met.

Advantages:

- Produces a dendrogram like agglomerative clustering.
- Can capture cluster structure differently, sometimes better than agglomerative for specific datasets.

Disadvantages:

- Computationally expensive, typically more than agglomerative clustering.
- Less commonly used due to complexity.

Applications of Hierarchical Clustering

- Gene expression analysis in bioinformatics.
- Document clustering for topic analysis.
- Image segmentation in computer vision.
- Social network analysis to detect communities.

---

4. Density-Based Clustering

Density-based clustering identifies clusters as dense regions of data points, separating areas of low density as noise or outliers. These methods are useful for detecting arbitrary-shaped clusters and handling noisy data.

4.1 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- Forms clusters by grouping closely packed points.
- **Points in sparse regions are treated as noise/outliers. Parameters:**

1. Eps (ε): Neighborhood radius to consider neighboring points.
2. MinPts: Minimum number of points required to form a dense region. Algorithm Steps:
1. Choose a point not yet visited.
2. If the point has ≥ MinPts within ε, form a new cluster.
3. Expand the cluster by recursively visiting all density-reachable neighbors.
4. Points that do not belong to any cluster are classified as noise. Advantages:

- Can find arbitrary-shaped clusters.
- Naturally identifies outliers/noise.
- **No need to specify the number of clusters beforehand. Disadvantages:**
- Sensitive to choice of ε and MinPts.
- Struggles with clusters of varying densities.
- Not suitable for high-dimensional data without preprocessing.

4.2 OPTICS (Ordering Points To Identify the Clustering Structure)

- An extension of DBSCAN.
- Instead of forming explicit clusters, it produces a reachability plot that orders points by density.
- Can handle clusters with varying densities, which DBSCAN cannot do easily.
- Useful for visualizing cluster structure and extracting clusters at different density thresholds.

Other Density-Based Variants

- HDBSCAN (Hierarchical DBSCAN): Builds a hierarchical clustering based on density and automatically selects the best clusters.
- DENCLUE: Uses a density function to assign points to clusters.

Applications

- Geospatial analysis: identifying densely populated regions.
- Anomaly detection: fraud detection, network intrusion detection.
- Image processing: clustering pixels with similar intensity or color.
- Market analysis: identifying customer groups in transaction data.

---

5. Evaluation of Clustering Results

Evaluating clustering is challenging because clustering is unsupervised, and no predefined labels exist. Evaluation can be internal (based on data alone) or external (compared to ground truth).

5.1 Internal Measures

Use only the dataset to assess clustering quality.

1. Silhouette

Coefficient (S(i)) [

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

]

- (a(i)) = average distance from point i to other points in its cluster (intra-cluster distance).
- (b(i)) = average distance from point i to points in the nearest cluster (nearest inter-cluster distance).
- Range: −1 to 1, higher = better clustering.

2. **Dunn Index:**

- Ratio of minimum inter-cluster distance to maximum intra-cluster distance.
- Higher values indicate well-separated and compact clusters.

3. **Davies-Bouldin Index:**

- Average similarity ratio of each cluster with its most similar cluster.
- Lower values indicate better clustering.

5.2 External Measures

Require ground truth labels to compare clustering results.

1. **Rand Index (RI):**

- Measures agreement between clustering and true labels.
- Range 0–1, higher = better.

2. **F-Measure:**

- Harmonic mean of precision and recall for clusters vs true labels.

---

6. Outlier Detection and Handling

An outlier is a data point that deviates significantly from the majority of the data.

Outlier detection is important for robust clustering and analysis.

Types of Outliers

1. **Global Outliers:**
   - Points far from the overall dataset.
   - Example: A transaction of ₹1,000,000 among typical ₹1,000 transactions.
2. **Contextual Outliers:**
   - Outliers in a specific context.
   - Example: 30°C is normal in summer but abnormal in winter.
3. **Collective Outliers:**
   - Group of points behaving differently from the rest.
   - Example: sudden spike in network traffic from a subset of IP addresses.

---

1. **Statistical Methods:**
    - Z-score, Grubbs' test, assuming underlying distribution.
2. **Distance-Based Methods:**
    - k-Nearest Neighbor (k-NN) distance; points far from neighbors are outliers.
3. **Density-Based Methods:**
    - LOF (Local Outlier Factor) identifies points in sparse regions compared to neighbors.
4. **Clustering-Based Methods:**
    - Points not belonging to any cluster (e.g., DBSCAN noise points).

Handling Outliers

- Remove: if caused by errors or noise.
- Transform data: e.g., logarithmic or square root scaling.
- **Use robust algorithms:**
    - K-Medoids instead of K-Means.

- DBSCAN or density-based clustering to detect and isolate outliers.

1. K-Means Clustering

Objective: Minimize **Sum of Squared Errors (SSE)** within clusters.

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

- C_i = cluster i
- x = data point in cluster
- μ_i = centroid of cluster i

Example:

Data points: {2, 4, 6, 8}, k = 2

- Initial centroids: 2 and 6
- Cluster assignment → C1={2,4}, C2={6,8}
- New centroids: μ1 = (2+4)/2 = 3, μ2 = (6+8)/2 = 7
- SSE = $(2-3)^2 + (4-3)^2 + (6-7)^2 + (8-7)^2 = 4$

2. K-Medoids (PAM)

Objective: Minimize total distance from points to **medoid** (actual data point).

$$Cost = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, m_i)$$

- m_i = medoid of cluster i
- $d(x, m_i)$ = distance (Euclidean/Manhattan)

Example:

Points: {1,2,3,10}, k=2

- Medoids: 2 and 10
- Cluster assignment → C1={1,2,3}, C2={10}
- Total cost: |1−2| + |2−2| + |3−2| + |10−10| = 1+0+1+0 = 2

3. Silhouette Coefficient

Measures clustering quality (internal evaluation).

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

- $a(i)$ = avg. distance to points in **same cluster**
- $b(i)$ = avg. distance to points in **nearest other cluster**

Example:

Point i → a(i)=2, b(i)=5

$$S(i) = \frac{5 - 2}{\max(2, 5)} = \frac{3}{5} = 0.6$$

→ Good clustering (closer to 1)

UNIT-V: Data Mining Applications and Tools

Web Mining

Definition:

Web Mining is the process of applying data mining techniques to discover patterns, trends, and useful knowledge from the World Wide Web. It extracts valuable insights from web data, which may include web content, web structure, and web usage information.

Types of Web Mining

Web Content Mining

Focuses on extracting useful information from the actual contents of web pages such as text, images, audio, video, and structured records (tables, metadata).

Example: Google indexing web pages, extracting product reviews from e-commerce sites, text classification.

Web Structure Mining

Uses graph theory to analyze the hyperlink structure of the web. Each web page is a node, and hyperlinks are edges.

Helps in understanding the relationship between different websites or pages.

Example: PageRank algorithm used by Google to rank pages, identifying communities on social media.

Web Usage Mining

Focuses on analyzing user interaction data collected from web server logs, cookies, browser history, and clickstreams.

Helps in predicting user behavior and personalization.

Example: Amazon and Netflix providing personalized recommendations, website optimization based on click behavior.

Applications of Web Mining

Search Engines: Ranking and retrieving relevant web pages (Google, Bing).

Recommender Systems: Personalized product/movie recommendations (Amazon, Netflix, YouTube).

Web Personalization: Customizing website content for individual users.

Customer Profiling: Understanding customer behavior, segmentation, and targeted advertising.

Fraud Detection: Detecting abnormal browsing or purchasing behavior.

Business Intelligence: Market trend analysis and decision-making support.

Advantages

Extracts hidden knowledge from large-scale web data.

Improves user experience through personalization.

Enhances marketing and customer relationship management (CRM).

Supports decision-making and business intelligence.

Helps in fraud detection and security monitoring.

Disadvantages

Privacy concerns due to user data collection and tracking.

Handling large and unstructured web data is complex.

Dynamic nature of web pages makes extraction difficult.

Ethical issues in user profiling and targeted advertising.

Requires high computational resources.

---

2. ## Text Mining

Definition:

Text Mining (or Text Data Mining) is the process of extracting meaningful patterns, knowledge, and insights from large amounts of unstructured text data using natural language processing (NLP), machine learning, and statistical techniques.

Steps in Text Mining

1. **Text Preprocessing**
   - *Tokenization*: Splitting text into words/tokens.
   - *Stop-word removal*: Removing common words (is, the, and).
   - *Stemming & Lemmatization*: Reducing words to their root/base form.
2. **Feature Extraction**
   - Bag-of-Words (BoW) model.
   - TF-IDF (Term Frequency–Inverse Document Frequency).
   - Word embeddings (Word2Vec, GloVe, BERT).
3. **Pattern Discovery**
   - Classification (spam vs. ham emails).
   - Clustering (grouping news articles).
   - Topic modeling (LDA, NMF).

Applications

- Spam filtering (Gmail, Outlook).
- Sentiment analysis (Twitter, customer reviews).
- Document summarization (news aggregation).
- Chatbots & Virtual Assistants (Alexa, Siri, ChatGPT).
- Legal & medical text analysis.

Advantages

- Handles vast unstructured text data.
- Improves decision-making with text analytics.
- Automates classification and summarization.
- Enhances customer engagement

(chatbots, reviews). Disadvantages

- Ambiguity in natural language (context problems).
- Requires preprocessing and high computational power.
- Difficulties in handling multiple languages & slang.
- Privacy and ethical issues in analyzing personal text data.

---

3. **Spatial**

Data

Mining

Definition:

Spatial Data Mining is the process of discovering interesting patterns and relationships from spatial data such as maps, satellite images, GPS data, and sensor records.

Examples of Tasks

- Finding clusters of disease outbreaks in specific regions.
- Detecting traffic patterns in smart cities.
- Mining satellite images for agriculture, deforestation, or climate monitoring.

Applications

- GIS (Geographic Information Systems): Mapping and land use analysis.
- Remote Sensing: Environmental and climate monitoring.
- Urban Planning: Road/transport design, smart city development.

- Disaster Management: Flood, earthquake, wildfire prediction & response.
- Healthcare: Epidemic spread analysis.

Advantages

- Provides location-aware insights.
- Supports disaster preparedness and management.
- Enhances agricultural and climate studies.
- Helps in resource allocation in smart cities.

Disadvantages

- Spatial data is often huge and computationally expensive.
- Accuracy depends on quality of satellite/GPS data.
- Privacy concerns in location-based services.
- Complex integration with non-spatial data.

4. Temporal and Sequence Data Mining

Definition:

- Temporal Data Mining: Extracting patterns from time-dependent data.
- Sequence Data Mining: Identifying ordered patterns in sequential datasets.

Examples

- Temporal: Stock price trends, weather forecasting, sensor data analysis.
- Sequence: Market basket sequence (laptop → bag → accessories), DNA sequence analysis.

Applications

- Weather forecasting and climate modeling.
- Stock market prediction.
- Bioinformatics (DNA/protein sequence analysis).
- Customer purchase behavior analysis.
- Predictive maintenance in IoT and industry.

Advantages

- Captures time-dependent trends and patterns.
- Supports forecasting and decision-making.
- Applicable in diverse domains (finance, healthcare, retail). Disadvantages

- Seasonal/periodic variations make predictions complex.
- Sensitive to noise and outliers.
- High computational requirements.

5. Introduction to Big Data and Scalable Mining Big Data

Definition:

Big Data refers to datasets that are too large, fast-changing, or complex for traditional data processing tools.

Characteristics (5 V's):

- Volume: Massive data size.
- Velocity: Speed of data generation.
- Variety: Structured, semi-structured, unstructured data.
- Veracity: Reliability and quality issues.
- Value: Useful insights extracted.

Scalable Data Mining:

Refers to algorithms and tools that efficiently process large-scale datasets using distributed and parallel computing.

- Frameworks: Hadoop (MapReduce), Apache Spark, Flink.
- Techniques: Distributed clustering, parallel classification, streaming analytics.

Applications

- Real-time fraud detection in banking & e-commerce.
- Large-scale recommender systems (Netflix, YouTube, Amazon).
- Social media analytics (Facebook, Twitter trends).
- Healthcare big data (genomics, patient records).
- Smart cities & IoT analytics.

Advantages

- Handles extremely large and diverse datasets.
- Enables real-time analysis and decision-making.
- Improves accuracy of recommendations and predictions.
- Scalable for cloud and distributed environments.

Disadvantages

- High infrastructure and storage cost.
- Data privacy and security issues.
- Requires specialized skills and frameworks.
- Complexity in managing and cleaning massive data.

6. Data Mining Tools

6.1 WEKA (Waikato Environment for Knowledge Analysis) Definition:

WEKA is an open-source data mining tool developed at the University of Waikato, New Zealand. It is written in Java and provides a collection of machine learning algorithms and data preprocessing tools. Features:

- Supports classification, clustering, regression, association rule mining, and visualization.
- Provides both GUI for easy use and API for developers.
- Supports data preprocessing, feature selection, and model evaluation.
- **Can handle ARFF, CSV, and other data formats. Applications:**

- Research and academic teaching in machine learning.
- Experimenting with classification/clustering algorithms.

- **Rapid prototyping of**

predictive models.

Advantages:

- Open-source and free.
- Easy-to-use GUI, suitable for beginners.
- Large collection of algorithms.

- **Active research**

community support.

Disadvantages:

- Limited scalability for very large datasets.
- Slower performance compared to big data frameworks.

- **Less suitable for**

production deployment.

Conclusion:

WEKA is an excellent tool for beginners, researchers, and educators to learn and experiment with machine learning and data mining techniques.

6.2 RapidMiner

Definition:

RapidMiner is a data science platform that provides advanced data mining, machine learning, and deep learning capabilities. It offers both open-source and commercial enterprise editions.

Features:

- Drag-and-drop visual workflow designer (no programming required).
- Supports machine learning, text mining, and big data integration.
- Built-in operators for data preprocessing, modeling, and evaluation.
- **Can be extended with**

Python and R scripting.

Applications:

- Enterprise analytics and business intelligence.
- Fraud detection, customer churn prediction.
- Predictive maintenance in industries.
- **Big data and**

text analytics.

Advantages:

- User-friendly interface with minimal coding.
- Scales well for enterprise applications.
- Integration with big data tools (Hadoop, Spark).
- **Strong support for**

advanced analytics.

Disadvantages:

- Some advanced features require a paid enterprise license.
- Resource-intensive (requires powerful hardware for large datasets).
- **Less flexible than pure**

coding-based solutions. Conclusion:

RapidMiner is a powerful tool for both beginners and professionals, widely used in businesses for predictive analytics and enterprise-level data mining.

6.3 **O**

ra

ng

e

De

fin

itio

n:

Orange is an open-source data mining and machine learning tool written in Python. It is known for its interactive, widget-based workflows and easy visualization features.

Features:

- Simple drag-and-drop workflow design using widgets.
- Built-in tools for classification, clustering, regression, and text mining.
- Strong visualization support (scatter plots, heatmaps, decision trees).
- **Can be extended with**

Python scripts.

Applications:

- Education and training in data science.
- Exploratory data analysis (EDA).
- Text mining and bioinformatics research.
- **Rapid model**

prototyping.

Advantages:

- Beginner-friendly and visually interactive.
- Lightweight and easy to install.
- Integrates with Python ecosystem (NumPy, Pandas, Scikit-learn).
- **Good visualization for**

quick insights.

Disadvantages:

- Not ideal for very large datasets.
- Limited compared to enterprise tools like RapidMiner.
- Fewer advanced features than WEKA or Spark ML.

7. Case Studies in Data Mining

7.1 Business Intelligence

Definition:

Business Intelligence (BI) refers to the use of data mining, analytics, and visualization tools to support decision-making in organizations.

Role of Data Mining in BI:

- Extracts hidden patterns from sales, finance, and customer data.
- Helps in forecasting trends and improving operational efficiency.
- Provides managers with actionable insights for strategy and planning. Example:
  - Walmart uses data mining for inventory management, demand forecasting, and dynamic pricing strategies.

Applications:

- Sales forecasting.
- Customer segmentation.
- Risk management.
- **Supply chain optimization.**

Advantages:

- Improves decision-making and efficiency.
- Enhances competitiveness.
- **Reduces costs by optimizing resources.**

Disadvantages:

- High cost of BI tools and infrastructure.
- Requires skilled analysts.
- **Risk of data misinterpretation.**

Conclusion:

Data mining strengthens BI by turning raw data into strategic knowledge, enabling organizations to remain competitive.

7.2 **Fraud Detection Definitio n:**

Fraud Detection involves using data mining and anomaly detection techniques to

identify unusual or suspicious activities in financial transactions, insurance claims, and online activities.

Role of Data Mining in Fraud Detection:

- Uses clustering, classification, and outlier detection to spot abnormal patterns.
- **Identifies risky transactions in real-time.**

Example:

- Credit card companies detect unusual spending patterns (e.g., sudden purchases in a foreign country).
- Banks use fraud detection systems to block unauthorized transactions. Applications:
  - Banking and financial services.
  - Insurance claim verification.
  - Cybersecurity (phishing/spam detection).
- **E-commerce fraud prevention.**

Advantages:

- Reduces financial losses.
- Enhances customer trust.
- **Real-time fraud monitoring.**

Disadvantages:

- May produce false positives (legitimate transactions flagged).
- Requires continuous updates due to evolving fraud tactics.
- High implementation and maintenance costs.

Conclusion:

Data mining plays a crucial role in fraud detection by identifying anomalies, protecting financial institutions, and ensuring security.

7.3 **E-commerce**

Definition:

E-commerce companies use data mining to enhance customer experience, increase sales, and optimize marketing strategies.

Role of Data Mining in E-commerce:

- Analyzes customer browsing and purchasing behavior.
- Builds recommendation systems using collaborative filtering and association rule mining.
- Performs market basket analysis to identify product combinations frequently purchased together.

Example:

- Amazon, Flipkart use recommendation engines to suggest products.
- Cross-selling and upselling strategies based on market basket analysis. Applications:
- Personalized product recommendations.
- Customer behavior analysis.
- Dynamic pricing strategies.
- **Inventory and supply chain management.**

Advantages:

- Increases sales and customer satisfaction.
- Improves targeting and marketing efficiency.
- **Strengthens customer loyalty.**

Disadvantages:

- Privacy concerns in customer data collection.
- May mislead customers with irrelevant recommendations.
- Requires large-scale infrastructure.

8. **Ethical and Privacy**

Issues in Data Mining

Introduction:

While data mining offers powerful insights for businesses, healthcare, finance, and government, it raises significant ethical and privacy concerns. These issues revolve around data ownership, misuse of personal information, and fairness in algorithmic decisions.

Key Concerns

1. **Privacy Concerns**
   - Mining personal data without consent can violate individual privacy.
   - Example: Collecting browsing history or purchase data without informing users.
2. **Data Ownership**
   - Unclear responsibility over who owns data — individuals, organizations, or governments.
   - Example: Social media platforms claiming ownership of user-generated content.
3. **Bias in Data Mining**
   - Algorithms trained on biased datasets may reinforce gender, racial, or social biases.
   - Example: Hiring algorithms discriminating against certain groups.
4. **Ethical Issues**
   - Use of mined data for mass surveillance, targeted advertising, or political manipulation.
   - Example: Cambridge Analytica scandal (Facebook data used for political campaigns).

Solutions

- Data Anonymization: Removing personal identifiers before mining.
- Transparent Algorithms: Making AI/ML models explainable and fair.
- Privacy-Preserving Data Mining (PPDM): Techniques like homomorphic encryption, differential privacy to protect sensitive data.
- Legal Frameworks: Regulations such as GDPR (General Data Protection Regulation, EU) and HIPAA (Health Insurance Portability and Accountability Act, USA) to enforce responsible data use.

Advantages of Ethical Practices

- Builds user trust and loyalty.
- Ensures compliance with international laws.
- Reduces misuse of personal and sensitive data.
- Promotes fairness in AI-driven decision-making. Disadvantages if Ignored
- Loss of reputation and customer trust.
- Legal penalties and financial losses.
- Social harm due to discrimination and misinformation.

1. Web Mining

1.1 PageRank Formula (Web Structure Mining)

$$PR(A) = (1-d) + d \sum_{i \in M(A)} \frac{PR(i)}{L(i)}$$

- PR(A) = PageRank of page A
- d = damping factor (typically 0.85)
- M(A) = set of pages linking to A
- L(i) = number of outbound links from page i

Example:

- Page A has inbound links from B (PR=0.4, L=2) and C (PR=0.2, L=1), d=0.85

$$PR(A) = 0.15 + 0.85 \left(\frac{0.4}{2} + \frac{0.2}{1} \right) = 0.15 + 0.85(0.2 + 0.2) = 0.15 + 0.34 = 0.49$$

1.2 Click-Through Rate (Web Usage Mining)

$$CTR = \frac{\text{Number of clicks on a link}}{\text{Number of times link shown}} \times 100$$

Example:

- Link displayed 1000 times, clicked 50 times → CTR = (50/1000)×100 = 5%

2. Text Mining

2.1 Term Frequency (TF)

$$TF(t,d) = \frac{\text{Number of times term t appears in document d}}{\text{Total terms in document d}}$$

2.2 Inverse Document Frequency (IDF)

$$IDF(t) = \log \frac{N}{1 + n_t}$$

- N = total number of documents
- n_t = number of documents containing term t

2.3 TF-IDF

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t)$$

Example:

- Term "data" appears 3 times in a doc of 100 words. Appears in 5 of 50 documents.

$$TF = 3/100 = 0.03, \quad IDF = \log(50/5) = \log(10) \approx 1 \Rightarrow TF - IDF = 0.03 \times 1 = 0.03$$

3. Spatial Data Mining

3.1 Euclidean Distance (for clustering spatial points)

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Example:

- Point A(2,3), B(5,7) → $d = \sqrt{(5-2)^2 + (7-3)^2} = \sqrt{9+16} = \sqrt{25} = 5$

3.2 Density of points

$$Density = \frac{\text{Number of points in region}}{\text{Area of region}}$$

Example:

- 50 trees in 100 m^2 → Density = 50/100 = 0.5 trees/m^2

---

4. Temporal & Sequence Data Mining

4.1 Support in Sequence Mining

$$Support(X \to Y) = \frac{\text{Number of sequences containing X followed by Y}}{\text{Total number of sequences}}$$

Example:

- 20 customers; 5 buy laptop then mouse → Support = 5/20 = 0.25

4.2 Confidence in Association Rule

$$Confidence(X \to Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X)}$$

Example:

- Support(laptop)=8/20, Support(laptop ∪ mouse)=5/20 → Confidence = (5/20)/(8/20) = 0.625 = 62.5%

---

5. Big Data & Scalable Mining

5.1 MapReduce Basic Formula

$$\text{Map: } (k_1, v_1) \rightarrow \text{list}(k_2, v_2) \quad ; \quad \text{Reduce: } (k_2, list(v_2)) \rightarrow (k_3, v_3)$$

Example: Word count

- Map: ("data",1), ("data",1), ("mining",1) → group by key
- Reduce: ("data",[1,1]) → ("data",2), ("mining",[1]) → ("mining",1)

5.2 Big Data Characteristics (5 V's)

- Volume, Velocity, Variety, Veracity, Value

6. Evaluation Metrics for Data Mining

6.1 Accuracy (Classification/Text Mining)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

Example:

- TP=80, TN=50, FP=10, FN=5 → Accuracy = (80+50)/(80+50+10+5) ×100 = 130/145 ×100 ≈ 89.7%

6.2 Precision & Recall

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}$$

Example:

- TP=80, FP=10, FN=5 → Precision=80/(80+10)=0.888, Recall=80/(80+5)=0.941