**1MIDSEM 1 QP**

**process of knowledge discovery**

The **Knowledge Discovery in Databases (KDD) process** consists of multiple steps that transform raw data into useful insights. These steps include **data preprocessing, data mining, pattern evaluation, and knowledge representation.**

# Data Cleaning

* + Data cleaning removes **noisy and irrelevant** data.
  + Deals with **missing values** by filling, interpolating, or removing incomplete records.
  + **Noisy data**, which contains random variations/errors, is cleaned to improve data quality.
  + Techniques used:

# Data discrepancy detection

* + - **Data transformation tools** to standardize formats.

# Data Integration

* + Combines **heterogeneous data** from multiple sources into a unified dataset.
  + Sources include databases, data warehouses, web data, and other repositories.
  + Integration methods:
    - **Data Migration tools** move data between systems.
    - **Data Synchronization tools** ensure data consistency across sources.
    - **ETL (Extract, Transform, Load) process** extracts data, transforms it, and loads it into a data warehouse.

# Data Mining: Concepts and Techniques

1. **Data Selection**
   * Selects relevant data for analysis.
   * Techniques for data selection:
     + **Neural Networks** – AI-based models for feature selection.
     + **Decision Trees** – Hierarchical decision-making for choosing data.
     + **Naïve Bayes** – Probabilistic method for data selection.
     + **Clustering & Regression** – Methods to identify patterns in selected data.

# Data Transformation

* + Converts raw data into a format suitable for mining.
  + Steps in data transformation:
    - **Data Mapping** – Assigns source elements to destination elements for transformation.
    - **Code Generation** – Creates transformation programs to process data.

# Data Mining

* + The actual process of extracting useful patterns from data.
  + Transforms **task-relevant data into patterns.**
  + Determines the model’s purpose using **classification, characterization, clustering, and association rule mining.**

# Pattern Evaluation

* + Identifies meaningful patterns from mined data.
  + Determines the **interestingness score** of each pattern.
  + Uses techniques like **summarization and visualization** to make patterns more understandable.

# Knowledge Representation

* + Uses visualization tools to present results.
  + Methods:
    - **Reports**
    - **Tables**
    - **Rules (Discriminant rules, Classification rules, Characterization rules)**
  + **Iterative Process**: The evaluation and mining steps can be refined, new data can be integrated, and transformations can be adjusted to improve results.

# Preprocessing of Databases

Steps **1 to 4** (Data Cleaning, Integration, Selection, and Transformation) form **data preprocessing**, which prepares data for mining.

The **Data Mining step** interacts with:

# Users

* + **Knowledge Bases**

Extracted patterns may be stored for future reference or used for further processing.

# Data Sources for Data Mining

Data can be mined from various sources, such as:

* + **Databases**
  + **Data Warehouses**
  + **Web Data**
  + **Streaming Data** from IoT or live sources

# Key Challenges and Issues in Data Mining (KDD Issues)

1. **Human Interaction**
   * Users must define objectives, select algorithms, and interpret results.
   * Automated tools cannot replace expert decision-making.

# Overfitting

* + A model may fit training data too well but fail on new data.
  + Requires techniques like **cross-validation** to avoid poor generalization.

# Outliers

* + **Outliers** are extreme values that deviate from normal patterns.
  + They can skew results or reveal hidden insights.

# Interpretation

* + Extracted patterns need to be interpretable and meaningful.
  + Complex models (e.g., deep learning) may be hard to explain.

# Visualization

* + Presenting data in an understandable format is crucial.
  + Visualization tools help users grasp complex patterns.

# Large Datasets

* + Processing huge volumes of data requires efficient storage and computing techniques.

# High Dimensionality

* + Datasets with many attributes (features) are challenging to analyze.
  + **Feature selection** techniques help reduce dimensions.

# Multimedia Data

* + Mining from images, videos, and audio requires specialized techniques.

# Missing Data

* + Data with missing values can lead to inaccurate models.
  + Techniques like **imputation** or **data interpolation** address this.

# Irrelevant Data

* + Unimportant features add noise and reduce efficiency.
  + **Feature selection** removes unnecessary attributes.

# Noisy Data

* + Random errors affect the quality of mined patterns.
  + Noise reduction techniques like **smoothing** or **outlier detection** help.

# Changing Data

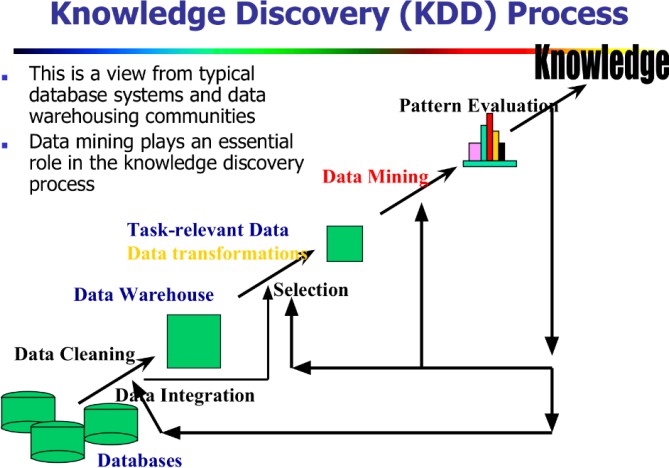
* + Data is constantly updated, requiring **real-time mining** techniques.

# Data Integration

* + Integrating different data sources is challenging due to differences in formats, storage systems, and access protocols.

# Applications of Data Mining

* + Business Intelligence
  + Healthcare Analytics
  + Fraud Detection
  + Recommendation Systems
  + Market Analysis
  + Social Media Mining
  + Scientific Research



# How data mining is applied for data warehouse What is a Data Warehouse?

A **Data Warehouse** is a **central repository** of integrated data from multiple, diverse sources.

It is designed to support **analysis, reporting, and decision-making** by storing data in a

**structured, summarized**, and **historical** form.

µ-· "' ¯' **Real-World Scenario: AllElectronics Example**

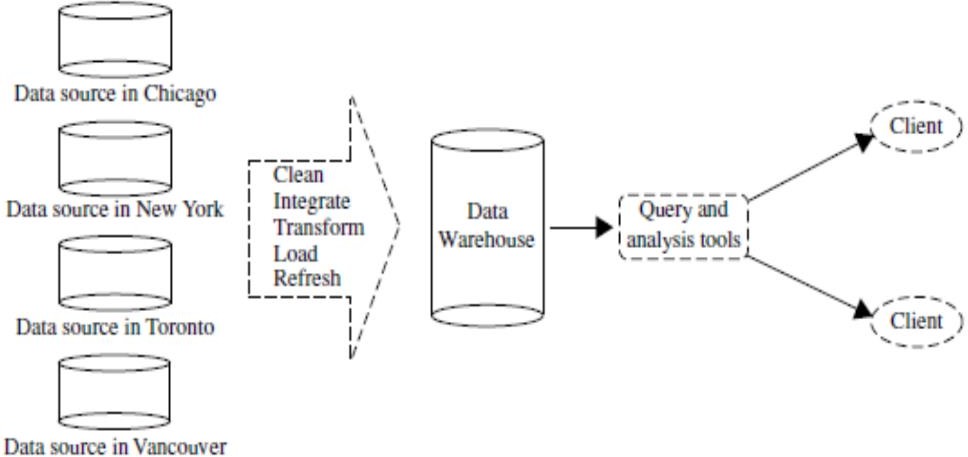
Imagine **AllElectronics** is a **global company** with branches in different countries. Each branch maintains **its own database**, possibly in different formats and locations.

Now the **CEO** wants a report:

“Company’s sales per item type per branch for the third quarter.” This is **very difficult** using operational databases:

* + Data is **scattered**.
  + Data may be in **different formats**.
  + Merging and analyzing that data in real-time is inefficient.

But if **AllElectronics had a data warehouse**, this report would be **quick and easy**.



^–¹\_−¯ **How Is a Data Warehouse Built?**

The construction involves the following **ETL process**:

1. **Data Cleaning** – Removing noise and inconsistencies.
2. **Data Integration** – Combining data from multiple sources.
3. **Data Transformation** – Converting data into suitable formats.
4. **Data Loading** – Moving the data into the warehouse.
5. **Data Refreshing** – Periodically updating data to keep it current.

••.‘c'–⬛ **Structure: Multidimensional Modeling**

A **data warehouse** is not just a giant table — it uses a **multidimensional model** called a

# data cube:

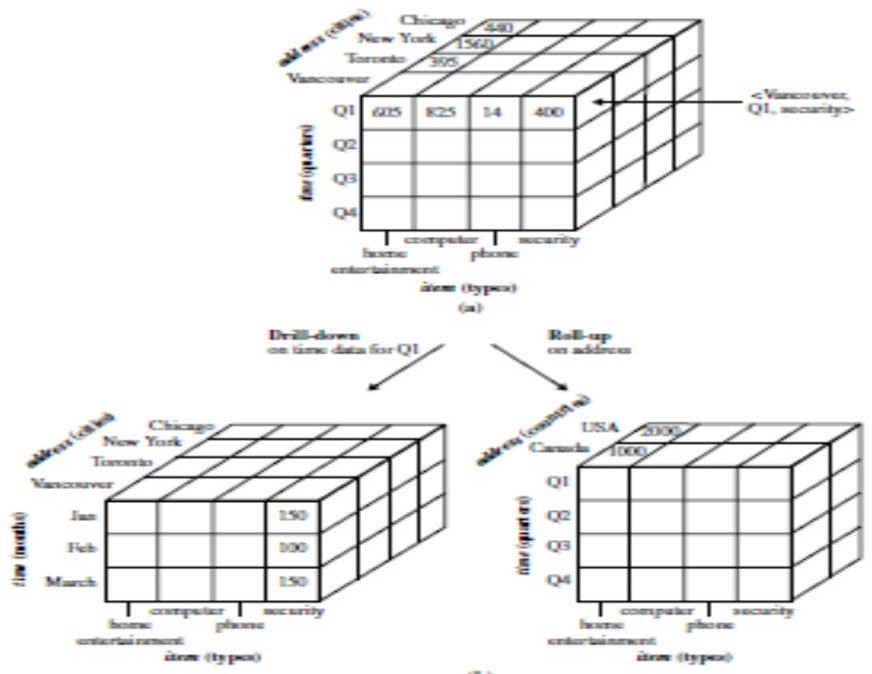
J u´,◆ **Data Cube Example**

For AllElectronics, the cube might have:

* + **Dimension 1: Location** (Chicago, New York, Toronto, Vancouver)
  + **Dimension 2: Time** (Q1, Q2, Q3, Q4)
  + **Dimension 3: Item Type** (Computer, Security, Phone, Home Entertainment)

Each **cell in the cube** contains a **precomputed aggregate value**, e.g., total sales amount. Example:

Cell (Vancouver, Q1, Security) = $400,000



# ’ ⬛ Operations in a Data Warehouse (OLAP)

OLAP = **Online Analytical Processing**

It allows users to **analyze data from multiple angles**. Key operations include:

# Drill-down

* + View more **detailed data**.
  + Example: From sales per **quarter** to sales per **month**.

# Roll-up

* + View more **summarized data**.
  + Example: From sales per **city** to sales per **country**.

These operations provide **flexibility** to analyze trends and make decisions.

−–¯,|⬛

# ⬛/# Why Data Warehouses Matter

* + **Fast decision-making** with historical and summarized data.
  + **Better business insight** from trends, patterns, and anomalies.
  + **Efficient data retrieval** for executives and analysts.
  + Essential for **Data Mining**, **BI (Business Intelligence)**, and **predictive analytics**.

**explain how classification and regression data mining functionaltiies are used to specify kinds of patterns in data mining tasks**

**Classification — *Predicting Categorical Outcomes***

# Definition:

Classification is the task of learning a model from data where the goal is to **predict a category or class label** for new or unseen data.

# How it specifies patterns:

* + It **analyzes training data** with known class labels and learns patterns that differentiate between different classes.
  + These patterns can be **expressed as rules, decision trees, or mathematical models**

that describe how different input features relate to class outcomes.

# Example patterns:

* + "IF a car has high gas mileage AND low engine power, THEN it belongs to the 'economy' class."
  + "IF temperature > 30°C AND humidity > 70%, THEN climate = 'tropical'."

# Common techniques:

* + Decision Trees
  + Naïve Bayes
  + Neural Networks
  + Rule-based classifiers
  + Support Vector Machines (SVM)

# Applications:

* + Spam email detection (spam vs. not spam)
  + Medical diagnosis (disease vs. no disease)
  + Customer segmentation (high-value vs. low-value customers)
* **Regression — *Predicting Numerical Outcomes***

# Definition:

Regression involves learning a model from data that can **predict a continuous (numeric) value**, rather than a category.

# How it specifies patterns:

* + It finds **relationships between input variables and a continuous output**, often in the form of linear or nonlinear equations.
  + These patterns define how changes in input features affect a numerical outcome.

# Example patterns:

* + Predicting house prices based on area, location, and number of rooms.
  + Estimating crop yield based on rainfall, soil type, and fertilizer usage.

# Common techniques:

* + Linear Regression
  + Logistic Regression (for classification tasks)
  + Polynomial Regression
  + Neural Networks (for numeric output)

# Applications:

* + Predicting stock prices
  + Forecasting weather
  + Estimating product demand

# Relevance to Pattern Specification in Data Mining Tasks

In data mining:

* + **Classification** helps identify **discrete patterns** (e.g., class boundaries).
  + **Regression** helps uncover **continuous patterns or trends** (e.g., growth curves). Both functionalities:
  + Are essential for **predictive modeling**.
  + May involve **feature selection or relevance analysis** to focus on the most informative attributes.
  + Help in **decision-making and trend analysis** by modeling patterns that capture the behavior of data.

**Applications of Data Mining**

*"Where there are data, there are data mining applications."*

Data mining plays a crucial role across various fields, and its applications are nearly limitless. Some of the most significant applications include **business intelligence** and **web search engines**, as well as specialized domains like **bioinformatics, medical research, and**

# software engineering.

1. **Business Intelligence (BI)**

Business Intelligence (BI) involves gathering, analyzing, and interpreting business data to enhance decision-making and competitive advantage.

# Why is Data Mining Important in BI?

Businesses need insights into:

* + **Customer behavior** – What products do customers prefer?
  + **Market trends** – What products/services are in demand?
  + **Supply chain optimization** – How can companies reduce costs?
  + **Competitor analysis** – What strategies are competitors using?

# Data Mining in BI Applications

Data mining is the backbone of BI, enabling businesses to:

* + **Analyze market trends** – Find emerging opportunities.
  + **Understand customer preferences** – Identify buying patterns.
  + **Improve decision-making** – Optimize pricing and inventory management.

# Examples of BI Applications Using Data Mining

|  |  |
| --- | --- |
| **Application** | **Data Mining Technique Used** |
| **Reporting & analytics** | Data warehousing, OLAP (Online Analytical Processing) |
| **Customer**  **segmentation** | Clustering (groups customers based on behavior) |
| **Fraud detection** | Anomaly detection, predictive modeling |
| **Market basket analysis** | Association rule mining (e.g., "customers who buy milk also buy bread") |
| **Predictive analytics** | Classification and regression |

For example, **Amazon** uses data mining to analyze user behavior, recommend products, and optimize pricing.

# Web Search Engines

A **search engine** is a system that retrieves information from the internet based on user queries. The process of search engine operation involves **crawling, indexing, and ranking** pages.

# How Data Mining Helps Search Engines

* + **Crawling** – Determines which pages to visit and how often.
  + **Indexing** – Decides which pages should be indexed.
  + **Ranking** – Uses algorithms to rank pages for relevance.
  + **Personalization** – Adjusts search results based on user behavior.

# Challenges in Web Search Engines

* + Handling **huge** amounts of data.
  + Dealing with **dynamic** and **fast-changing** web content.
  + Improving **search relevance** with minimal processing time.

# Popular Search Engine Algorithms Using Data Mining

|  |  |
| --- | --- |
| **Algorithm** | **Purpose** |
| **PageRank (Google)** | Ranks web pages based on the number of quality backlinks. |
| **HITS (Hyperlink-Induced Topic Search)** | Finds authoritative web pages and hubs (pages linking to many authoritative pages). |
| **Latent Semantic Indexing (LSI)** | Identifies relationships between terms to improve search results. |

Search engines like **Google and Bing** use data mining techniques to refine search results and enhance user experience.

# Other Applications of Data Mining

1. **Web Page Analysis**
   * **Web page classification** – Categorizing web pages (e.g., sports, news, entertainment).
   * **Clustering** – Grouping similar web pages for better search indexing.
   * **PageRank & HITS algorithms** – Determining the importance of a page based on links.

# Recommender Systems

* + **Used in platforms like Netflix, Amazon, and YouTube** to suggest products or content.
  + **Collaborative filtering** – Predicts user preferences based on similar users.
  + **Content-based filtering** – Suggests items similar to what a user has interacted with.

# Market Basket Analysis (Targeted Marketing)

* + **Finds correlations in purchasing patterns** (e.g., "customers who buy coffee also buy sugar").

# Used in supermarkets and e-commerce platforms.

* + **Techniques Used:** Association Rule Mining (e.g., Apriori algorithm).

# Medical and Biological Data Analysis

* + **Microarray data analysis** – Helps in genetic research and disease prediction.
  + **Biological sequence analysis** – Identifies patterns in DNA/protein sequences.
  + **Biological network analysis** – Maps interactions between genes/proteins.

Example: **AI-driven cancer detection** uses classification models trained on medical images.

# Data Mining in Software Engineering

* + **Bug prediction** – Identifies code with a high probability of containing bugs.
  + **Code clone detection** – Finds duplicate or similar code to improve efficiency.
  + **Software defect prediction** – Identifies software modules likely to fail.

# Example: Google and Microsoft use machine learning to detect software vulnerabilities.

1. **Data Mining in Enterprise Software**

Many enterprise software solutions include built-in data mining tools. Some major tools are:

* + **SAS Enterprise Miner** – Predictive analytics and data mining.
  + **Microsoft SQL Server Analysis Services (SSAS)** – Business intelligence and data mining.

**Oracle Data Mining (ODM)** – Advanced analytics for enterprise applications.

**Nominal and ordinal attributes:**

# Nominal Attributes – Detailed Explanation

Nominal attributes are **qualitative or categorical** attributes that **represent names, labels, or categories**. These attributes **do not have any inherent order or ranking** among them — each value is just **a distinct category**, and **you cannot say one value is greater or lesser than another**.

# ⬛ Key Characteristics of Nominal Attributes:

|  |  |
| --- | --- |
| **Property** | **Description** |
| **Type** | Qualitative (categorical) |
| **Ordering** | No natural or meaningful order or ranking |
| **Arithmetic operations** | Not applicable (you can’t add, subtract, average, etc.) |
| **Comparison** | Equality or difference (== or !=) only |
| **Used in** | Classification, labeling, grouping |
| **Encoding for ML** | One-Hot Encoding or Label Encoding (if needed for numeric algorithms) |

˛\*C **Why They Matter in Data Mining**

In data mining and machine learning, it's essential to **identify nominal attributes** so that they are **processed properly**:

* + **Wrong assumption** of order or numeric value can lead to incorrect patterns or models.
  + Helps in **deciding preprocessing techniques**, especially for classification tasks.

‘ l\_H µ\_µ'– **Examples Explained**

# Hair Color → {black, brown, grey, red, white}

* + There’s **no natural order** (black is not greater than brown, etc.).
  + These are **labels**, simply identifying categories.

# Marital Status → {single, married, divorced}

* + No ranking (divorced is not greater or less than single).
  + These values only **categorize** a person’s relationship status.

# Occupation → {doctor, engineer, teacher}

* + Occupations are **distinct roles**, not numerically or ordinally related.

# ID Numbers / Zip Codes:

* + Though these may look like numbers, they are **not numerical in nature**.
  + Example: Zip code 90210 is not greater than 10001 in any meaningful way.
  + These are **identifiers** or **codes**, not values used for arithmetic.

# Ordinal Attributes – Detailed Explanation

**Ordinal attributes** represent **categorical data** where the **categories have a meaningful order or ranking**, but the **differences between the categories are not measurable or precisely defined**.

They are a middle ground between **nominal attributes** (no order) and **numeric attributes**

(ordered with measurable differences).

# ⬛ Key Characteristics of Ordinal Attributes

|  |  |
| --- | --- |
| **Feature** | **Description** |
| **Type** | Categorical (qualitative) |
| **Order** | Yes — values have a clear, meaningful ranking |
| **Distance between values** | No — we can't quantify the exact difference between categories |

|  |  |
| --- | --- |
| **Feature** | **Description** |
| **Arithmetic operations** | Not meaningful (can’t subtract or average them) |
| **Comparisons** | Only comparisons like greater than or less than make sense |
| **Typical Use** | Preference levels, rankings, ratings, scales |

‘ \_l\_Hµµ'– **Examples Explained**

# Size → {small, medium, large}

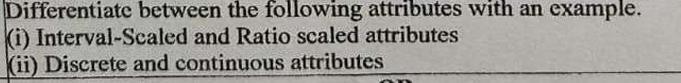
* + The order is meaningful: large > medium > small.
  + But we can’t say “large is 2x medium” or find how many units larger it is.

# Grades → {A, B, C, D, F}

* + A > B > C > D > F.
  + But the **gap** between A and B may not be the same as between C and D.

# Army Ranks → {Private, Sergeant, Captain, General}

* + Clear hierarchy: General outranks Captain, etc.
  + But the **difference in responsibility or experience** is not a fixed unit.



# Difference between Interval-Scaled and Ratio-Scaled Attributes

|  |  |  |
| --- | --- | --- |
| **Feature** | **Interval-Scaled Attribute** | **Ratio-Scaled Attribute** |
| **Definition** | Measures differences between values, but **no true zero** | Measures differences and has a  **meaningful zero** point |
| **Zero Point** | Arbitrary (zero doesn’t mean absence of quantity) | Absolute zero (zero = no quantity) |
| **Mathematical Operations** | Addition and subtraction allowed | All arithmetic operations allowed (add, subtract, multiply, divide) |
| **Example** | Temperature in Celsius or Fahrenheit | Weight, Height, Age, Income |

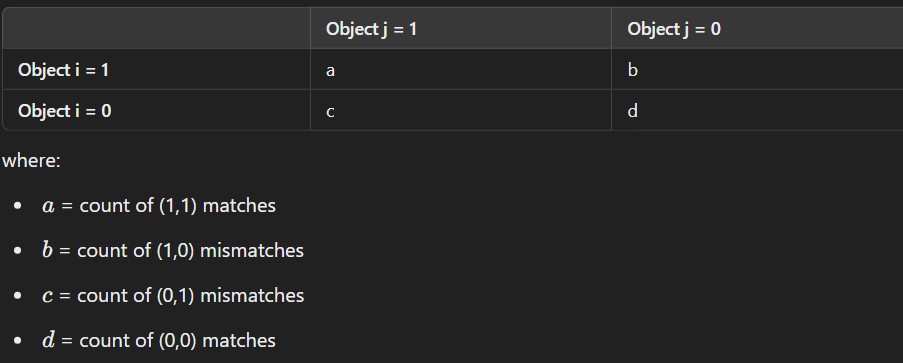
|  |  |  |
| --- | --- | --- |
| **Feature** | **Interval-Scaled Attribute** | **Ratio-Scaled Attribute** |
| **Ratio Meaningful?** | No (e.g., 20°C is not "twice as hot" as 10°C) | Yes (e.g., 20 kg is twice as heavy as 10 kg) |

1. **Difference between Discrete and Continuous Attributes**

|  |  |  |
| --- | --- | --- |
| **Feature** | **Discrete Attribute** | **Continuous Attribute** |
| **Definition** | Takes **finite or countable**  number of values | Takes **infinite** number of values within a range |
| **Value Type** | Whole numbers (often integers) | Real numbers (fractions, decimals allowed) |
| **Examples** | Number of students, cars, books | Temperature, height, weight, time |
| **Graph**  **Representation** | Dots on a number line | Continuous curve on a graph |
| **Measurement** | Counted | Measured |

**Proximity Measure for Binary Attributes**

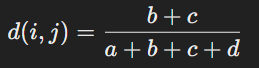
Binary attributes have only **two possible values** (e.g., Yes/No, 1/0, True/False). A **contingency table** is used to organize binary data for similarity calculations:



# Distance Measures:

1. **Symmetric Binary Distance:**

Used when both states (0 and 1) are equally important.



# Asymmetric Binary Distance:

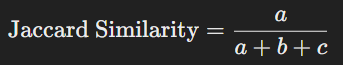
Used when one state (e.g., 1 = "Yes") is more significant than the other (e.g., 0 = "No").



Here, **d (0,0) matches are ignored** because the presence of a feature is more important than its absence.

# Jaccard Coefficient (For Asymmetric Binary Variables):

Used when **only "1" (positive matches) matter**, and "0" values are irrelevant.



This is commonly used in text mining and clustering.

# Dissimilarity Between Binary Variables - Example

Consider a dataset with **Gender** and other binary attributes:

* **Gender** is **symmetric** (both male and female are equally important).
* The other attributes (e.g., symptoms of a disease: Yes/No) are **asymmetric**. We represent:

# Y (Yes) and P (Present) as 1.

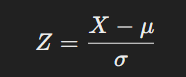
* **N (No) as 0**.

This helps compute distances appropriately.

# Standardizing Numeric Data

Since numeric data can have different scales, it is essential to **standardize** before measuring similarity.

# Z-score Standardization:

****

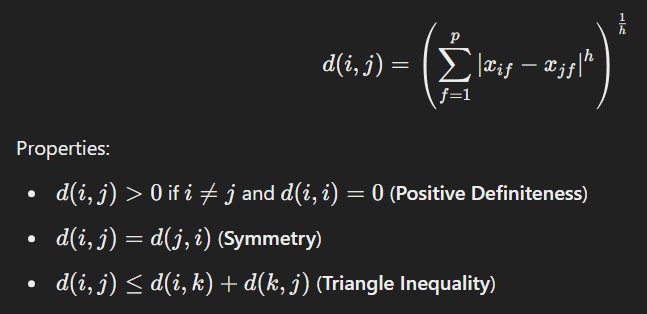
* 1. where:
     + XXX = raw data point
     + μ\muμ = mean
     + σ\sigmaσ = standard deviation
     + If Z>0Z > 0Z>0, the value is above the mean.
     + If Z<0Z < 0Z<0, the value is below the mean.

# Mean Absolute Deviation (MAD):

A more **robust** alternative to standard deviation, as it is less affected by extreme values.

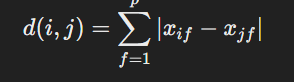
# Distance Measures for Numeric Data

The **Minkowski Distance** generalizes multiple distance measures:



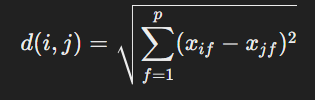
If a distance function satisfies these properties, it is called a **metric**. **Special Cases of Minkowski Distance:**

# Manhattan Distance (L1 norm, city-block distance)

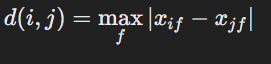
****

* + Example: Hamming distance (used in binary data).

# Euclidean Distance (L2 norm, most common measure)

****

**Supremum Distance (L∞ norm, Maximum norm)**



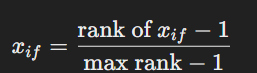
1. ​
   * Measures only the largest absolute difference.

# Ordinal Variables (Ranked Data)

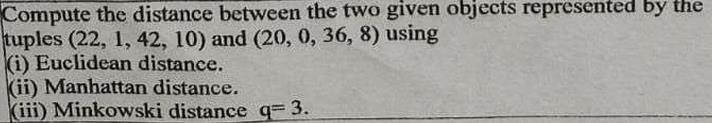
Ordinal variables have ordered categories, such as **Low, Medium, High** or **Rank 1, Rank 2, Rank 3**.

# Handling Ordinal Data:

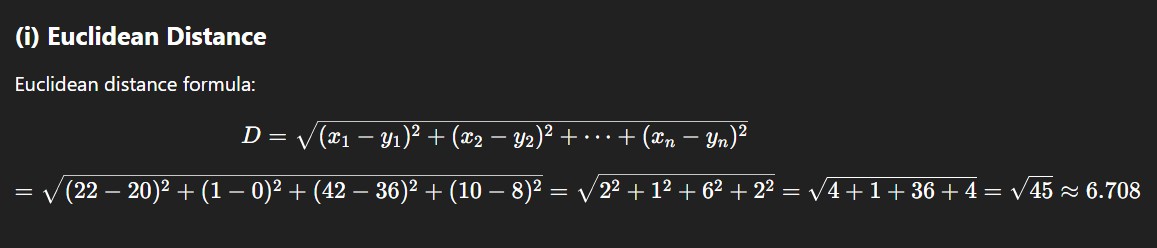
* + Assign **ranks** to the ordinal values.
  + Normalize values between **0 and 1** using:

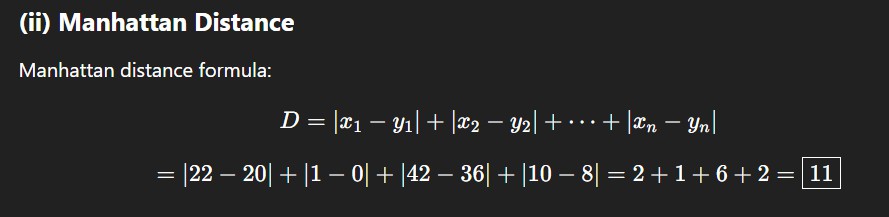


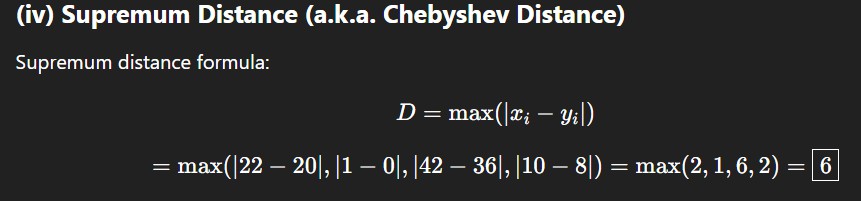
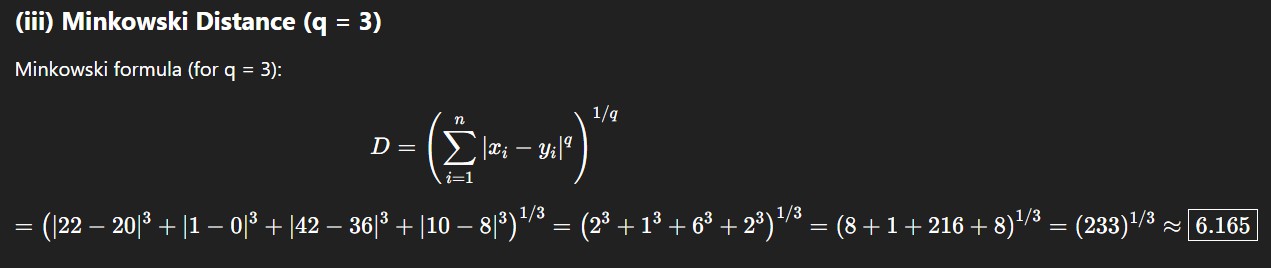
Then, compute distance using interval-based methods (e.g., Euclidean or Manhattan distance).

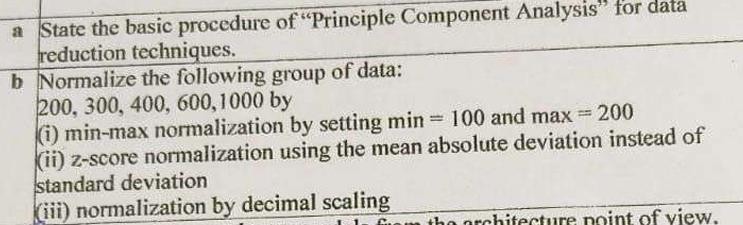


1. Supremum distance









# Principal Component Analysis (PCA) – Detailed Explanation

**Principal Component Analysis (PCA)** is a powerful statistical technique used primarily for **dimensionality reduction** while preserving as much of the variability in the data as possible. It transforms a large set of variables into a smaller one that still contains most of the information in the original set.

` '- **Why PCA is Needed**

In datasets with many attributes (dimensions), analysis can become complicated due to:

* + **Redundancy** (attributes being correlated),
  + **Noise**, and
  + **Increased computational cost** (curse of dimensionality).

PCA addresses this by reducing the number of dimensions without losing essential patterns or structures in the data.

’⬛ **Basic Steps of PCA**

Let’s now go through the **step-by-step procedure** of PCA:

# Data Normalization (Standardization)

Before performing PCA, we **normalize** the data so that each feature has **zero mean and unit variance**.

This ensures that:

* + All attributes are on the same scale,
  + Features with large ranges do not dominate the analysis.

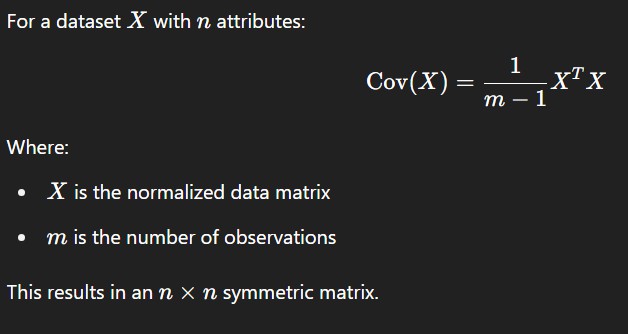
# Example:

If one attribute ranges from 1–10,000 and another from 0–1, the first will influence the results more unless normalized.

# Mathematically:

1. **Compute the Covariance Matrix**

The **covariance matrix** captures the **relationships (correlations)** between different attributes. It helps to understand how attributes vary with respect to each other.



# Compute the Eigenvectors and Eigenvalues

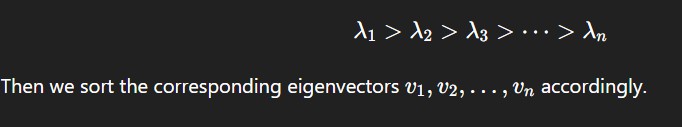
* + **Eigenvectors** of the covariance matrix represent the **principal components**.
  + **Eigenvalues** represent the **amount of variance** captured by each component.

So:

* + Each eigenvector points in the direction of maximum variance.
  + The eigenvalue tells us how important (significant) that direction is.

# Sort Eigenvectors by Eigenvalues

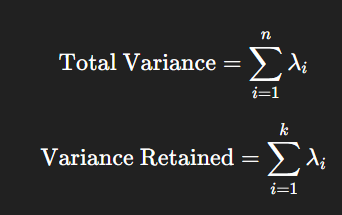
* + Eigenvectors are sorted in **descending order** of their eigenvalues.
  + The **first principal component** captures the most variance.
  + The **second component** captures the most of what’s left, and so on. Let’s say the eigenvalues are:



# Select the Top k Components

* + Decide how many dimensions you want to retain, say k, such that k≤n.
  + Often chosen by keeping the top components that account for **a desired percentage of total variance** (e.g., 95%).

Let:



# Transform the Original Data

Project the original normalized data onto the new k-dimensional subspace using the top k eigenvectors.



Where:

* + Y is the transformed data in lower dimensions,
  + X is the normalized original data,
  + W is the matrix with k eigenvectors (principal components) as columns.

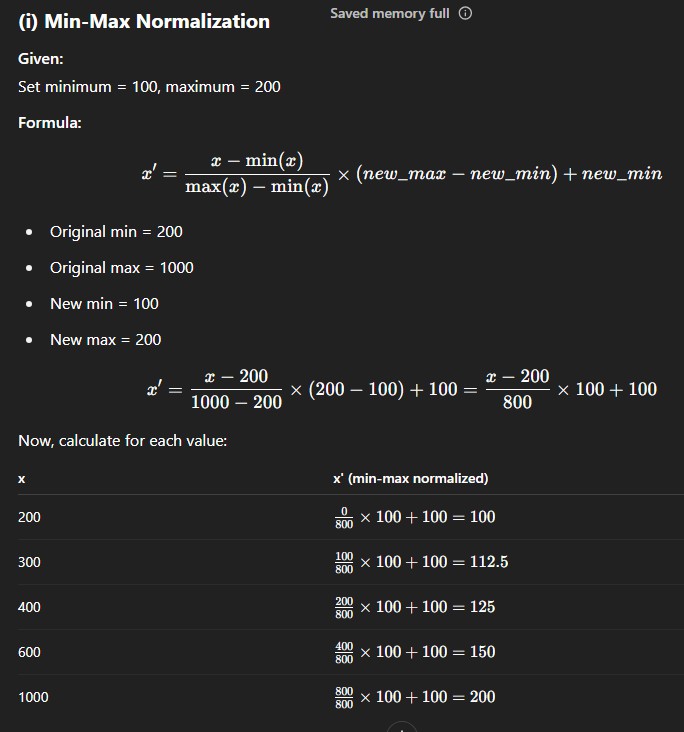
" ’◎´● **Benefits of PCA**

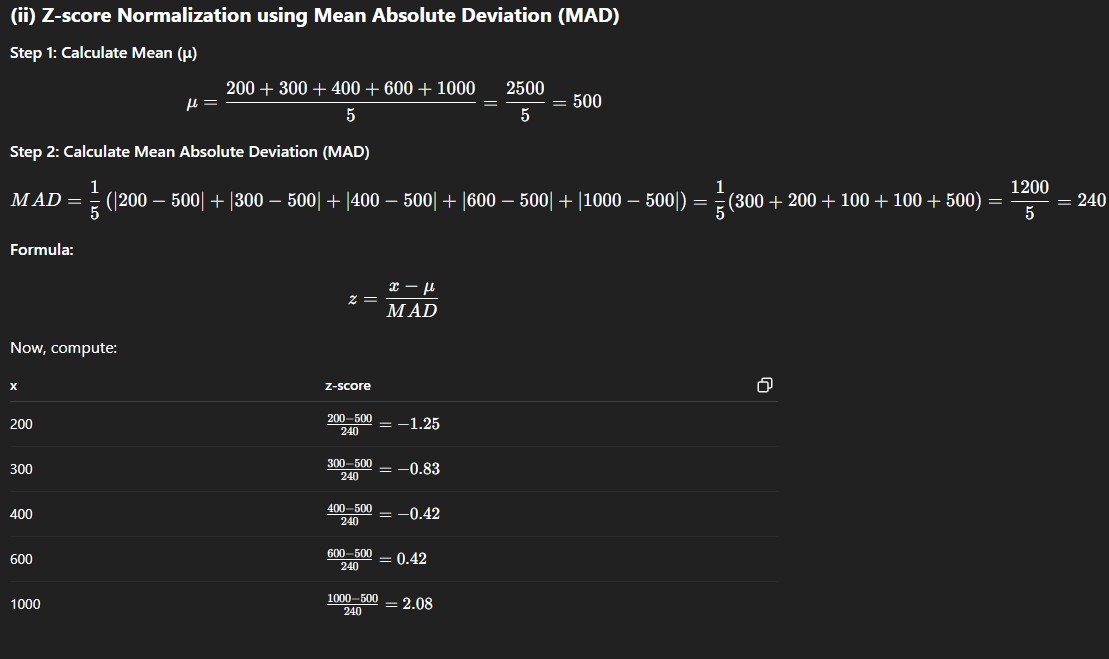
* + **Reduces dimensionality**: Removes less informative dimensions.
  + **Improves performance**: Less computation, faster processing.
  + **Removes noise**: Low-variance components are likely noise.
  + **Visualizes high-dimensional data**: In 2D/3D using top components.
  + **Uncovers structure**: Highlights patterns or clusters.

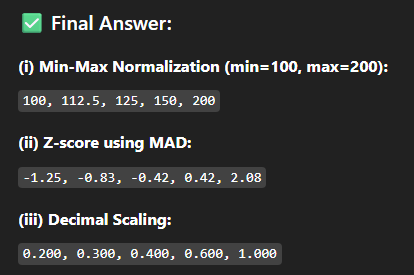
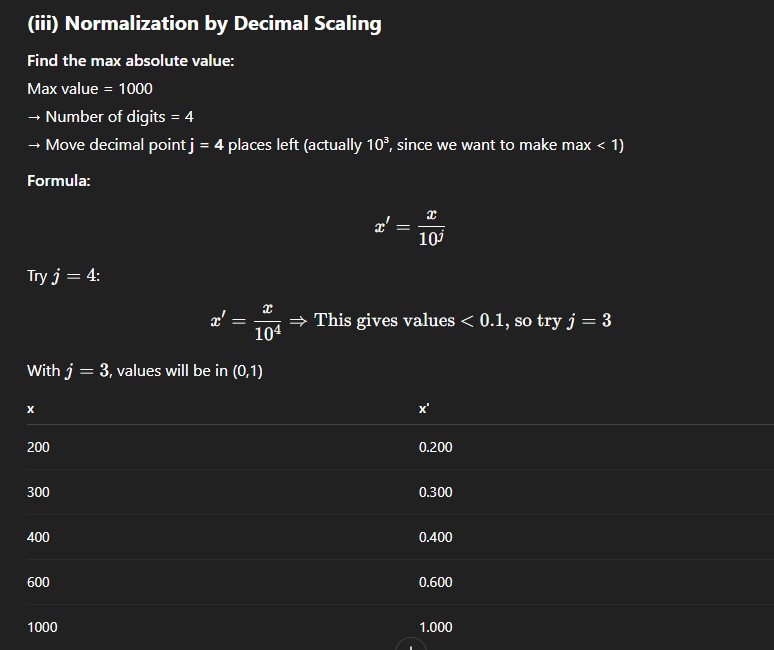
•Q˙ **Applications of PCA**

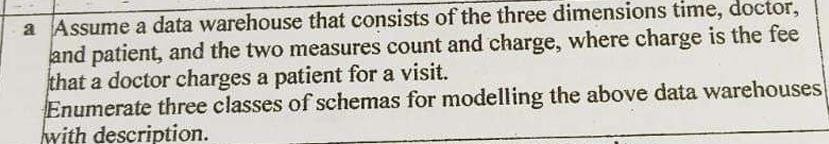
* + **Image compression**
  + **Face recognition**
  + **Pattern recognition**
  + **Data visualization**
  + **Preprocessing for machine learning models**

b)









# Star Schema

* + **Description:**

A central **fact table** contains the quantitative data (**count**, **charge**) and keys to each of the **dimension tables** (time, doctor, patient).

# Structure:

* + - **Fact Table:** Visit\_Fact(time\_id, doctor\_id, patient\_id, count, charge)

# Dimension Tables:

* + - * Time(time\_id, day, month, year)
      * Doctor(doctor\_id, name, specialty)
      * Patient(patient\_id, name, age, gender)

# Advantages:

* + - Simple to understand and use
    - Optimized for query performance

# Snowflake Schema

* + **Description:**

An extension of the star schema where dimension tables are **normalized** into multiple related tables.

# Structure:

* + - Similar fact table as in star schema
    - **Dimension Tables** are broken down:
      * Doctor(doctor\_id, name, specialty\_id)
      * Specialty(specialty\_id, specialty\_name)
      * Patient(patient\_id, name, gender\_id)
      * Gender(gender\_id, gender\_name)

# Advantages:

* + - Reduces data redundancy
    - More structured, saves storage

# Fact Constellation Schema (Galaxy Schema)

* + **Description:**

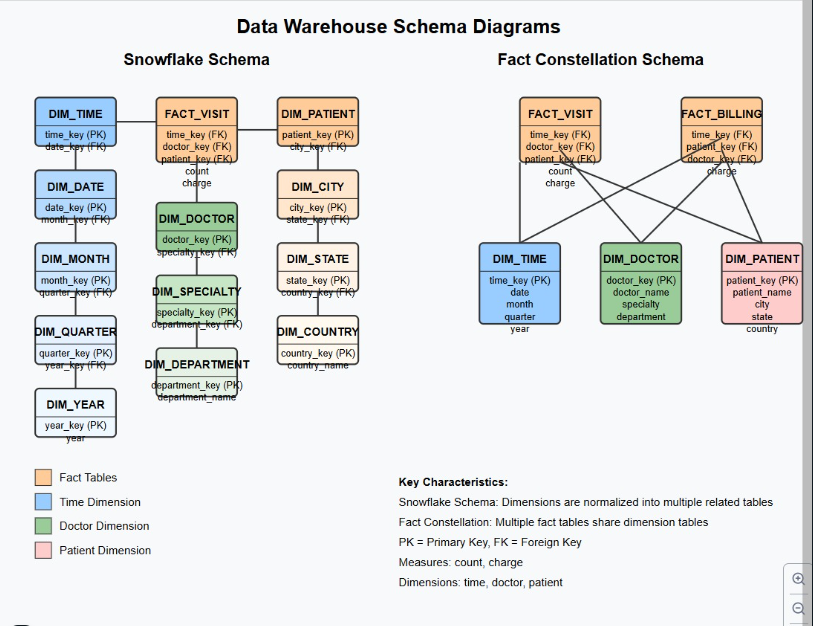
Multiple fact tables share dimension tables. Useful when there are **multiple processes or subject areas**.

# Structure:

* + - **Fact Tables:**
      * Visit\_Fact(time\_id, doctor\_id, patient\_id, count, charge)
      * Payment\_Fact(time\_id, patient\_id, amount\_paid)
    - **Shared Dimensions:** Time, Patient, Doctor

# Advantages:

* + - Supports complex queries across multiple facts
    - Promotes reuse of dimension tables



**give an overview about efficient data cube computation**

A **data cube** is a **multidimensional array of values** used to represent data along multiple dimensions, commonly used in **OLAP (Online Analytical Processing)**. Efficient computation of data cubes is essential for performance in large-scale data warehouses.

# Q˙• Why Efficient Data Cube Computation?

* + Data cubes can have **2^n** possible group-bys (cuboids) for **n dimensions**.
  + With large datasets and many dimensions, naive computation is **very expensive** in terms of **time and storage**.
  + Efficient techniques are needed to **reduce computation time**, **avoid redundancy**, and

# support fast query responses.

–'‘c•.• ⬛ **Key Concepts in Data Cube Computation**

# Cuboid:

* + A group-by of a particular subset of dimensions.
  + The full cube contains all possible cuboids.

# Lattice of Cuboids:

* + Cuboids form a lattice (a partially ordered set), where a parent cuboid can be used to compute child cuboids.

# Base Cuboid:

* + The original fact table with all dimensions.

# Apex Cuboid:

* + The cuboid with only aggregated data (no dimensions, just overall total).

# ◦O☼ Techniques for Efficient Data Cube Computation

1. **Multi-way Array Aggregation**
   * Partitions the cube into small chunks (multi-dimensional arrays).
   * Aggregates in multiple directions simultaneously.
   * Efficient for memory-resident data.

# Bottom-Up Computation (BUC)

* + Starts from the base cuboid and recursively computes more generalized cuboids.
  + Uses **pruning** to avoid computing unneeded cuboids.
  + Good for **sparse data**.

# Top-Down Computation (Star-Cubing)

* + Begins from the apex cuboid and drills down to compute detailed cuboids.
  + Uses **Star Tree structures** to efficiently access relevant aggregates.

# Iceberg Cubes

* + Computes only those cuboids whose **measure values exceed a threshold**.
  + Avoids materializing insignificant data (e.g., low-frequency combinations).

# Materialization Strategy

* + **Full Materialization**: All cuboids are computed (fast querying but storage-intensive).
  + **No Materialization**: Compute on-demand (saves space but slow queries).
  + **Partial Materialization**: A selected subset of useful cuboids are computed and stored based on **query workload**.

**Ways for handling missing data**

# Handling Missing Data in a Dataset – Explained with Examples

In real-world datasets, it’s common to encounter **missing values**. Handling them properly is crucial because missing data can:

* + Skew analysis results,
  + Reduce model accuracy,
  + Lead to incorrect conclusions.

# ⬛ Common Ways to Handle Missing Data

Here are the most widely used methods:

# Ignore the Tuple (Record Deletion)

Remove rows that have missing values.

* + ⬛ Best when:
    - The dataset is large.
    - The number of missing entries is very small.

 + Avoid if:

* + - Many rows have missing values — leads to data loss.

# Example:

Name | Age | Salary

Alice | 25 | 50000

Bob | | 45000 ← missing age Charlie | 30 | 60000

After removing:

Name | Age | Salary

Alice | 25 | 50000

Charlie | 30 | 60000

# Fill in with a Global Constant

Replace missing values with a fixed value like "Unknown" or 0.

* + ⬛ Simple and fast.

 + May add noise if the constant doesn’t make sense.

# Example:

text CopyEdit Original:

Name | Age | Salary

Alice | 25 | 50000 Bob | | 45000 Charlie | 30 | 60000

Filled with constant (e.g., Age = -1): Name | Age | Salary

Alice | 25 | 50000

Bob | -1 | 45000

Charlie | 30 | 60000

# Fill in with the Attribute Mean/Median/Mode

* + **Numerical data** → use **mean** or **median**.
  + **Categorical data** → use **mode** (most frequent category).

# Example (fill missing age with mean age):

Age values: 25, –, 30 → Mean = (25 + 30)/2 = 27.5

Updated:

Name | Age | Salary

Alice | 25 | 50000 Bob | 27.5| 45000

Charlie | 30 | 60000

# Fill in Using Class-Based Mean/Median

If data is grouped by class labels (like gender, department), fill missing values using the

# mean/median of that group. Example:

Department | Age

HR | 25

IT | –

HR | 27

→ Fill missing IT age using mean of IT department (if available)

# Predict the Missing Value Using a Model

Use a **machine learning model** (e.g., regression, decision tree) to **predict the missing value**

based on other attributes.

* + ⬛ More accurate than simple imputation.

 + More complex and computationally expensive.

# Example:

Train a regression model:

Features: [Experience, Education Level] Target: Age

→ Use it to predict missing ages.

# Use Data Interpolation (Time Series)

For **time series data**, use interpolation to estimate missing values between known points.

# Example:

Time | Temperature

1 | 30

2 | —

3 | 40

Interpolated value at time 2 = (30 + 40) / 2 = 35

**Handling Redundancy in Data Integration**

Redundancy is another major issue when integrating multiple data sources. It occurs when duplicate or unnecessary attributes exist in the combined dataset.

# Types of Redundancy

1. **Object Identification Redundancy**
   * The same object is given different names in different databases.
   * Example: Product\_ID vs. Item\_Number.

# Derivable Data

* + An attribute is redundant if it can be derived from other attributes.
  + Example: If **Annual Revenue** can be calculated using **Monthly Revenue**, storing both may be unnecessary.

# Detecting Redundancies

Redundant attributes can be identified using:

# Correlation Analysis

* **Covariance Analysis Correlation Analysis (Nominal Data)**

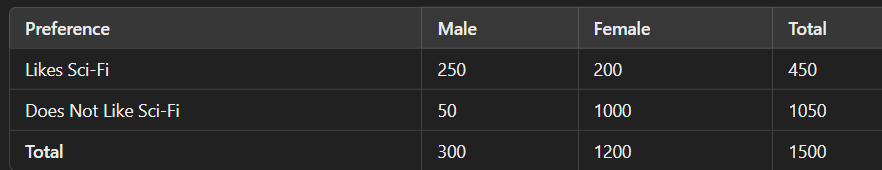
Correlation analysis helps determine how strongly one attribute is related to another. For

**nominal (categorical) data**, the **Chi-Square (Χ²) test** is used to find correlations.

# Chi-Square (Χ²) Test

The **Chi-Square test** checks whether two categorical attributes are statistically dependent.

# Example: Does Gender Influence Science Fiction Preference?

****

* The Chi-Square test can determine whether the **gender** (male/female) is correlated with liking **science fiction**.
* A higher **Χ² value** means a stronger correlation.

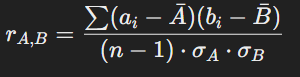
However, **correlation does not imply causation**. Example:

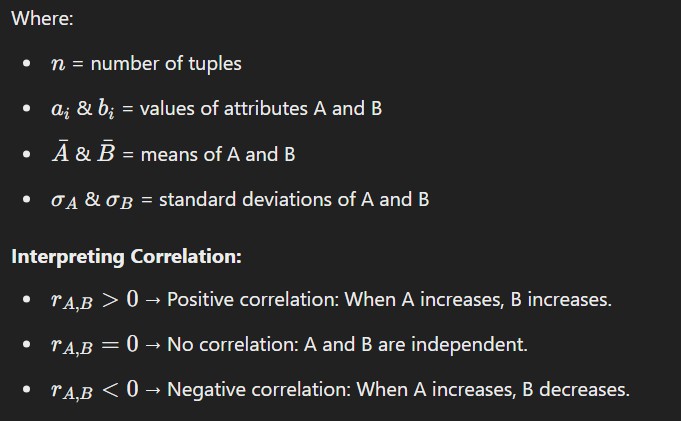
* The number of hospitals in a city and the number of car thefts may be correlated, but this does not mean hospitals cause car thefts.
* Instead, both may be linked to **a third factor: population size**.

# Correlation Analysis (Numeric Data)

For **numeric attributes**, correlation is measured using the **Correlation Coefficient (Pearson’s r)**.

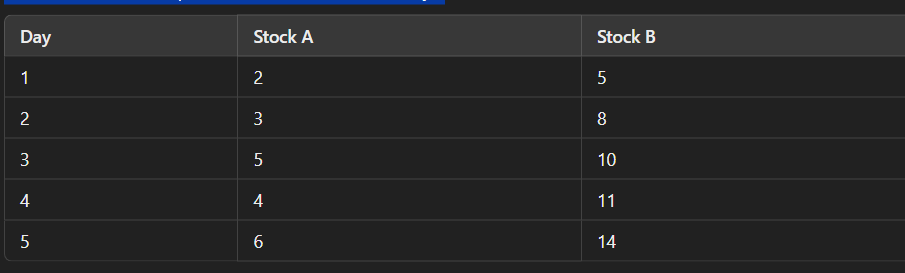
# Formula for Pearson’s Correlation Coefficient

****



**Example: Stock Price Correlation**

Consider two stock prices recorded over five days:



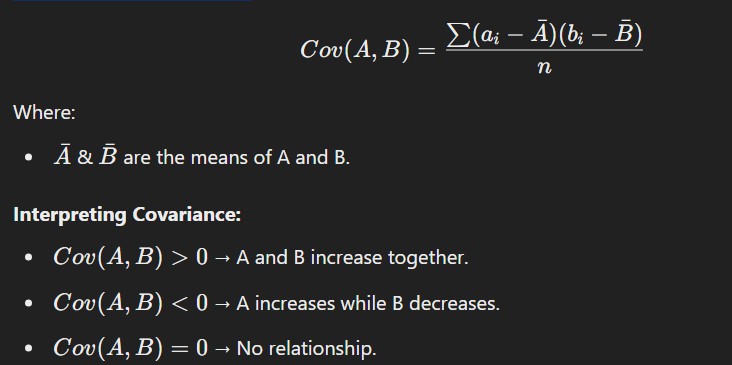
Using the correlation formula, we find r>0r > 0r>0, meaning **both stocks move together**. **Scatter Plots for Correlation**

* **Positive Correlation:** Points slope from **bottom left to top right**.
* **Negative Correlation:** Points slope from **top left to bottom right**.
* **No Correlation:** Points are scattered randomly.

# Covariance of Numeric Data

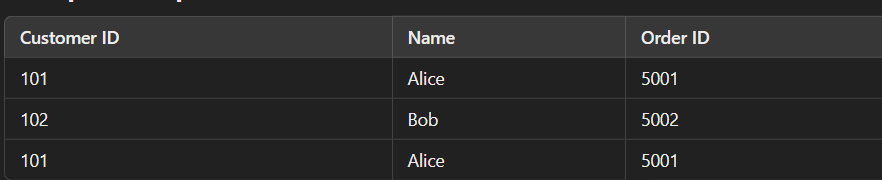
Covariance is another measure of how two numeric attributes change together.

# Covariance Formula:

****

**3.3.3 Tuple Duplication**

Besides detecting redundant attributes, **duplicate tuples** should also be identified and removed.



The first and last row are duplicates.

Removing duplicates **reduces storage** and **improves efficiency**.

**Data Warehouse Models**

A **Data Warehouse** is a centralized repository that stores integrated data from multiple sources to support **business intelligence (BI)** and **decision-making processes**. Data in a warehouse is structured, processed, and optimized for **querying and analysis** rather than transactional processing.

**Types of Data Warehouse Models**

Data warehouse models are classified based on their **architecture** and **organization** into three types:

1. **Enterprise Warehouse Model**
2. **Data Mart Model**
3. **Virtual Warehouse Model**
4. **Enterprise Warehouse Model**

The **Enterprise Data Warehouse (EDW)** is a **centralized** data repository that integrates data from multiple sources across an entire organization.

* + **Characteristics:**
    - Stores **historical and current data** from various departments.
    - **Highly structured** and follows a well-defined schema.
    - Supports **complex queries and reporting** for strategic decision-making.
    - Data is **processed using ETL (Extract, Transform, Load)** before being loaded.
  + **Advantages:**

⬛ Provides **a single version of truth** for the entire organization.

⬛ Ensures **data consistency and accuracy**.

⬛ Supports **large-scale data analysis** across departments.

* + **Disadvantages:**

+ **Expensive** and requires significant infrastructure.

+ **Complex to implement** and maintain.

+ **Slower response time** for queries due to its massive size.

* + **Example:**

A **multinational company** like Amazon uses an EDW to consolidate sales, inventory, customer behavior, and logistics data into a **single database** for **business analytics**.

1. **Data Mart Model**

A **Data Mart** is a **subset of an enterprise data warehouse** that focuses on a **specific business unit or department**, such as sales, finance, or marketing.

* + **Types of Data Marts:**

1. **Dependent Data Mart**
   * Extracts data from a centralized **Enterprise Data Warehouse**.
2. **Independent Data Mart**
   * Created separately without using an **Enterprise Data Warehouse**.
3. **Hybrid Data Mart**
   * Combines data from both **enterprise warehouses and external sources**.
   * **Characteristics:**
     + Smaller in scope and **department-specific**.
     + Easier and faster to implement than a full EDW.
     + Can be **built as a subset** of an enterprise data warehouse or independently.
   * **Advantages:**

⬛ **Faster access to data** for department-specific analysis.

⬛ **Lower cost** compared to an enterprise warehouse.

⬛ **Easier to maintain and scale**.

* + **Disadvantages:**

+ Data silos may form, leading to **inconsistencies**.

+ **Less comprehensive** compared to an enterprise data warehouse.

* + **Example:**

A **Retail Chain** might have separate **Data Marts** for:

* + - **Sales Department** (Tracks revenue, customer purchases, and product demand).
    - **Marketing Department** (Stores campaign data, customer engagement, and brand perception).

1. **Virtual Warehouse Model**

A **Virtual Data Warehouse** is an **on-demand view** of data derived from multiple sources, without physically storing all data in one location.

* + **Characteristics:**
    - **Does not physically store data** but provides **real-time access** to different sources.
    - Uses **middleware** and **data virtualization** techniques.
    - Works with **distributed databases** and cloud-based storage solutions.
  + **Advantages:**

⬛ Reduces **storage costs** as data is not physically duplicated.

⬛ Provides **real-time data access** and analysis.

⬛ Faster implementation compared to physical warehouses.

* + **Disadvantages:**

+ **Slower query performance** as data is retrieved in real-time.

+ **Limited historical data analysis** capabilities.

+ **Data inconsistency issues** if source systems change frequently.

* + **Example:**

A **financial institution** might use a **Virtual Data Warehouse** to fetch **real-time banking transactions**

from various databases without consolidating all data into a single location.

**OLAP Operations in Detail**

OLAP (Online Analytical Processing) is a technology that enables users to analyze multidimensional data interactively from different perspectives. It is used in data warehouses and business intelligence systems to provide quick insights by performing operations on data cubes.

**Concept Hierarchies in OLAP**

Concept hierarchies define multiple levels of data abstraction in OLAP. For example, a *time* dimension can have a hierarchy like:

**Day → Month → Quarter → Year**

A *location* dimension can have a hierarchy like:

**Street → City → State → Country**

Using concept hierarchies, OLAP provides flexibility for users to explore data at different levels of granularity.

**Typical OLAP Operations**

OLAP operations allow users to manipulate and analyze data cubes dynamically. The key operations include:

1. **Roll-up (Drill-up)**
2. **Drill-down (Roll-down)**
3. **Slice and Dice**
4. **Pivot (Rotate)**
5. **Drill-across**
6. **Drill-through**

Each of these operations helps in understanding business data in different ways.

1. **Roll-up (Drill-up) Definition**

Roll-up is an OLAP operation that aggregates data by:

* + Climbing up a **concept hierarchy**
  + Reducing the number of dimensions (Dimension Reduction)

**Example**

Consider a sales data cube with three dimensions: **Location, Time, and Product**.

* + Suppose the data is initially grouped by **City** (New York, Chicago, Toronto).
  + A roll-up operation can aggregate the data to a higher level, such as **Country** (USA, Canada).

**Another Example (Dimension Reduction)**

* + Suppose a data cube contains **Sales by Location and Time**.
  + If we remove the **Time** dimension, the cube will now show **total sales by Location** only.

– − *Roll-up provides a summarized view of data by reducing details.*

1. **Drill-down (Roll-down) Definition**

Drill-down is the reverse of roll-up. It navigates from summarized data to more detailed data by:

* + Stepping **down a concept hierarchy**
  + Introducing **additional dimensions Example**

Consider a sales cube where data is grouped by **Year**.

* + A drill-down operation can break the data into **Quarters → Months → Days** for a more detailed view.

**Another Example (Introducing New Dimensions)**

* + Suppose a sales cube contains only **Location and Time** dimensions.
  + A drill-down operation can introduce a **Customer** dimension, allowing analysis of sales by customer groups.

−– *Drill-down helps in detailed analysis and understanding finer data points.*

1. **Slice and Dice Slice Operation**

The **Slice** operation extracts a subcube by selecting a **single dimension value**. **Example**

* + If we have a sales cube with dimensions **Time, Location, and Product**, we can slice the cube for a specific value:
    - **Time = "Q1"**
    - This results in a new subcube showing only sales for **Quarter 1** across all locations and products.

**Dice Operation**

The **Dice** operation selects a subcube by applying conditions on **two or more dimensions**. **Example**

* + If we apply the conditions:
    - **Location = ("Toronto" OR "Vancouver")**
    - **Time = ("Q1" OR "Q2")**
    - **Product = ("Electronics" OR "Computers")**
  + The resulting subcube will contain only sales data that meet all these conditions.

−– *Slice filters data based on one dimension, while Dice filters based on multiple dimensions.*

1. **Pivot (Rotate) Definition**

Pivot (also called Rotate) changes the perspective of data by **rotating the axes** of a cube. This helps in better visualization.

**Example**

* + If a report initially shows **Sales by Product and Location**, a pivot can swap axes to show **Sales by Location and Product** instead.
  + A 3D cube can be transformed into a **series of 2D tables** for easier analysis.

– − *Pivoting helps in reorienting data for better readability and analysis.*

1. **Drill-Across Definition**

Drill-across is used when queries need to involve multiple **fact tables**. **Example**

* + Suppose a company maintains two fact tables:
    - **Sales** (containing sales data by region and time)
    - **Marketing Campaigns** (containing ad expenditure by region and time)
  + A drill-across operation can combine data from both tables to analyze **the impact of marketing on sales**.

– − *Drill-across helps in comparing multiple fact tables with common dimensions.*

1. **Drill-Through Definition**

Drill-through allows users to access **detailed data from a lower level** of the data cube by querying the **original relational database**.

**Example**

* + If a high-level sales summary shows that **Q2 sales were low**, a drill-through operation can retrieve **transaction-level data** to find the exact sales records.

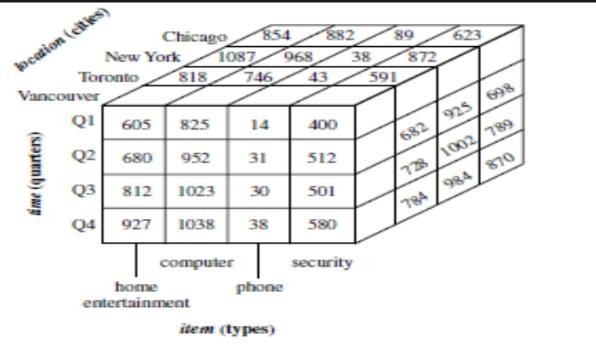
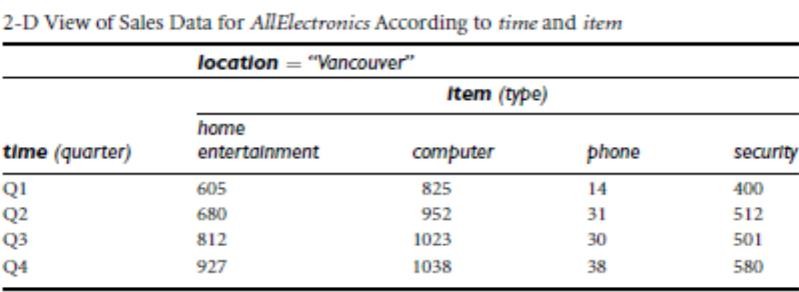
– − *Drill-through provides access to the most granular details stored in relational tables.*

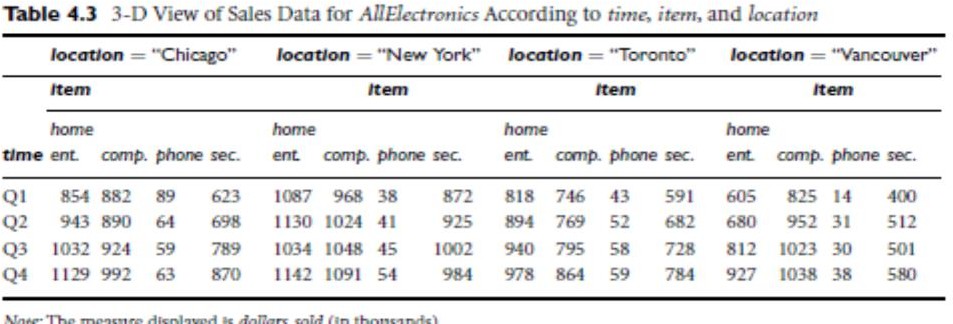
**Other OLAP Operations**

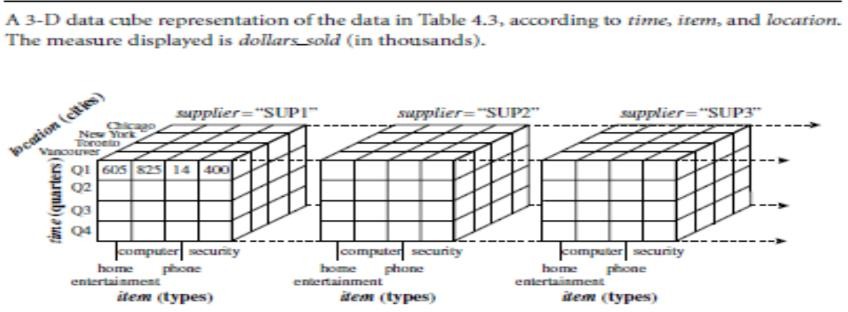
Some OLAP systems also support:

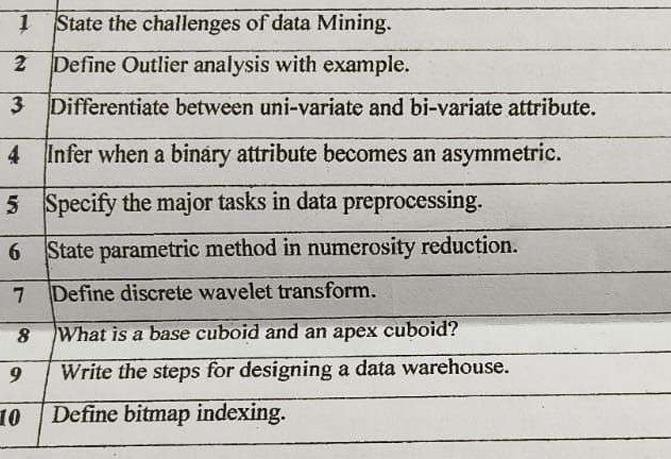
* + **Top-N Analysis:** Find the top-performing products/customers.
  + **Moving Averages:** Calculate trends over time.
  + **Currency Conversions:** Convert sales data to different currencies.

**Statistical Functions:** Compute growth rates, variances, and correlations









# State the challenges of data mining.

* + Handling large volumes of data
  + High dimensionality
  + Data quality issues (missing, noisy, inconsistent)
  + Integration from heterogeneous sources
  + Dynamic and evolving data
  + Data privacy and security
  + Scalability and efficiency of algorithms
  + Interpretation and visualization of results

# Define Outlier analysis with example.

**Outlier analysis** is the process of detecting data points that deviate significantly from the rest of the data.

**Example**: In a dataset of human heights, if most values range from 150 cm to 190 cm, a value of 300 cm is an outlier.

# Differentiate between uni-variate and bi-variate attribute.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Uni-variate Attribute** | **Bi-variate Attribute** |
| Definition | Involves a single variable | Involves two variables |
| Purpose | Analyzes distribution, mean, etc. | Analyzes relationships (correlation) |
| Example | Age, income | Age vs. income, height vs. weight |

1. **Infer when a binary attribute becomes an asymmetric.**

A **binary attribute** is **asymmetric** when the two states are **not equally important**, and one is considered more significant.

**Example**: In medical data, "has cancer" (1) vs. "does not have cancer" (0). Presence of cancer is more critical and needs more attention.

# Specify the major tasks in data preprocessing.

* + **Data cleaning** – Handling missing and noisy data.
  + **Data integration** – Combining data from multiple sources.
  + **Data transformation** – Normalization and aggregation.
  + **Data reduction** – Dimensionality and numerosity reduction.
  + **Data discretization** – Converting numeric data into categories.

# State parametric method in numerosity reduction.

A **parametric method** in numerosity reduction assumes that the data fits a known model and summarizes it using model parameters.

**Example**: Using **regression models** to fit data and represent it with model parameters (slope, intercept).

# Define discrete wavelet transform.

**Discrete Wavelet Transform (DWT)** transforms data into the wavelet domain, representing it as a set of coefficients to capture both **frequency** and **location**.

Used in data compression and feature extraction.

# What is a base cuboid and an apex cuboid?

* + **Base cuboid**: Contains the **most detailed data** with all dimensions.
  + **Apex cuboid**: Contains the **most generalized data**, without any dimensional values (total aggregation).

1. **Write the steps for designing a data warehouse.**
2. **Requirement analysis**
3. **Data source identification and integration**
4. **Data modeling (Star, Snowflake, or Fact Constellation schema)**
5. **ETL process (Extract, Transform, Load)**
6. **Designing metadata repository**
7. **Creating OLAP cubes**
8. **Implementing front-end tools for reporting**
9. **Testing and deployment**
10. **Define bitmap indexing.**

**Bitmap indexing** uses bitmaps (bit arrays) for each distinct value of an attribute, where each bit represents whether a row contains that value.

Efficient for attributes with **low cardinality** (e.g., gender, yes/no).