

# Machine Learning

Dr. Thyagaraju G S

# Course – Details

	<b>MACHINE LEARNING</b>	Semester	6
Course Code	<b>BCS602</b>	CIE Marks	50
Teaching Hours/Week (L: T:P: S)	4:0:0:0	SEE Marks	50
Total Hours of Pedagogy	50	Total Marks	100
Credits	04	Exam Hours	03
Examination type (SEE)	Theory		

# Course objectives:

1. To introduce the fundamental concepts and techniques of machine learning.
2. To understanding of various types of machine learning and the challenges faced in realworld applications.
3. To familiarize the machine learning algorithms such as regression, decision trees, Bayesian models, clustering, and neural networks.
4. To explore advanced concept like reinforcement learning and provide practical insight into its applications.
5. To enable students to model and evaluate machine learning solutions for different types of problems.

# Module1: Syllabus

**Introduction:** Need for Machine Learning, Machine Learning Explained, Machine Learning in Relation to other Fields, Types of Machine Learning, Challenges of Machine Learning, Machine Learning Process, Machine Learning Applications.

**Understanding Data - 1:** Introduction, Big Data Analysis Framework, Descriptive Statistics, Univariate Data Analysis and Visualization.

**Chapter-1, 2 (2.1-2.5)**

# Module2: Syllabus

**Understanding Data - 2:** Bivariate Data and Multivariate Data, Multivariate Statistics, Essential Mathematics for Multivariate Data, Feature Engineering and Dimensionality Reduction Techniques.

**Basic Learning Theory:** Design of Learning System, Introduction to Concept of Learning, Modelling in Machine Learning.

**Chapter-2 (2.6-2.8, 2.10), Chapter-3 (3.3, 3.4, 3.6)**

# Module3 : Syllabus

**Similarity-based Learning:** Nearest-Neighbor Learning, Weighted K-Nearest-Neighbor Algorithm, Nearest Centroid Classifier, Locally Weighted Regression (LWR).

**Regression Analysis:** Introduction to Regression, Introduction to Linear Regression, Multiple Linear Regression, Polynomial Regression, Logistic Regression.

**Decision Tree Learning:** Introduction to Decision Tree Learning Model, Decision Tree Induction Algorithms.

**Chapter-4 (4.2-4.5), Chapter-5 (5.1-5.3, 5.5-5.7), Chapter-6 (6.1, 6.2)**

# Module4 : Syllabus

**Bayesian Learning:** Introduction to Probability-based Learning, Fundamentals of Bayes Theorem, Classification Using Bayes Model, Naïve Bayes Algorithm for Continuous Attributes.

**Artificial Neural Networks:** Introduction, Biological Neurons, Artificial Neurons, Perceptron and Learning Theory, Types of Artificial Neural Networks, Popular Applications of Artificial Neural Networks, Advantages and Disadvantages of ANN, Challenges of ANN.

**Chapter-8 (8.1-8.4), Chapter-10 (10.1-10.5, 10.9-10.11)**

# Module5 : Syllabus

**Clustering Algorithms:** Introduction to Clustering Approaches, Proximity Measures, Hierarchical Clustering Algorithms, Partitional Clustering Algorithm, Density-based Methods, Grid-based Approach.

**Reinforcement Learning:** Overview of Reinforcement Learning, Scope of Reinforcement Learning, Reinforcement Learning as Machine Learning, Components of Reinforcement Learning, Markov Decision Process, Multi-Arm Bandit Problem and Reinforcement Problem Types, Model-based Learning, Model Free Methods, Q-Learning, SARSA Learning.

**Chapter -13 (13.1-13.6), Chapter-14 (14-1-14.10)**

# Course Outcomes (Course Skill Set)

At the end of the course, the student will be able to :

1. Describe the machine learning techniques, their types and data analysis framework.
2. Apply mathematical concepts for feature engineering and perform dimensionality reduction to enhance model performance.
3. Develop similarity-based learning models and regression models for solving classification and prediction tasks.
4. Build probabilistic learning models and design neural network models using perceptrons and multilayer architectures
5. Utilize clustering algorithms to identify patterns in data and implement reinforcement learning techniques

## **Module-1**

**Introduction:** Need for Machine Learning, Machine Learning Explained, Machine Learning in Relation to other Fields, Types of Machine Learning, Challenges of Machine Learning, Machine Learning Process, Machine Learning Applications.

**Understanding Data - 1:** Introduction, Big Data Analysis Framework, Descriptive Statistics, Univariate Data Analysis and Visualization.

**Chapter-1, 2 (2.1-2.5)**

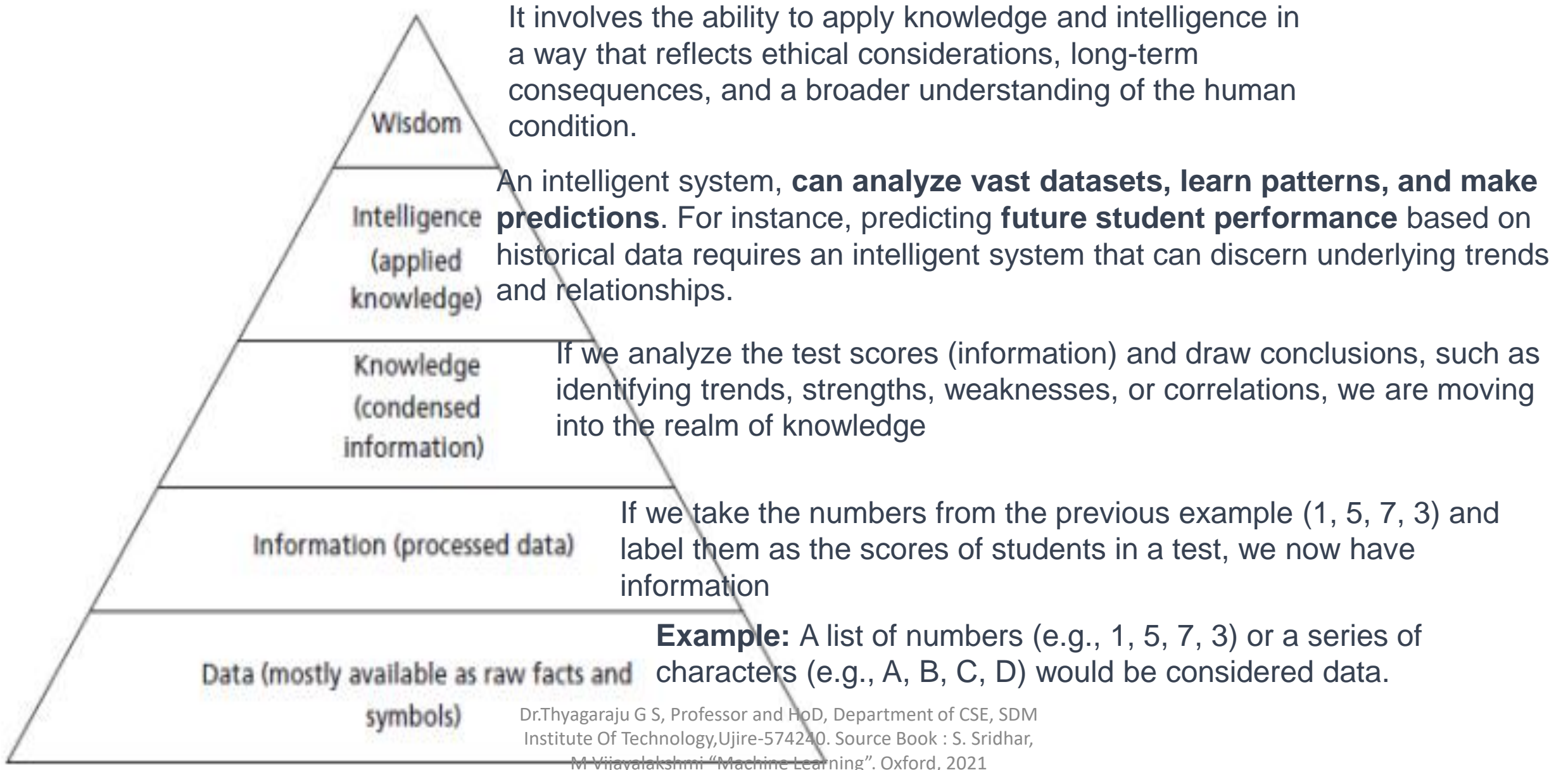
# Module1 : Part1 : Introduction

1. Need for Machine Learning,
2. Machine Learning Explained,
3. Machine Learning in Relation to other Fields,
4. Types of Machine Learning,
5. Challenges of Machine Learning,
6. Machine Learning Process,
7. Machine Learning Applications.

# Need for Machine Learning

1. Business Organization have numerous data
2. To analyze and extract the knowledge from the data stored in the various archives of the organization
  1. To facilitate decision making
  2. Useful for design new products
  3. Improve business processes and
  4. To develop decision support system

# Knowledge Pyramid



# Popularity of Machine Learning

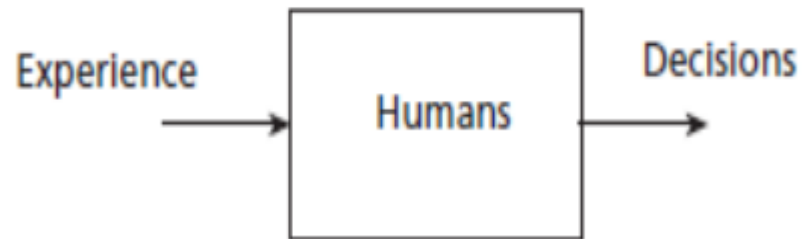
1. High Volume of Available data to manage
2. The Cost of storage has reduced
3. Availability of Complex Algorithms

# What is Machine Learning ?

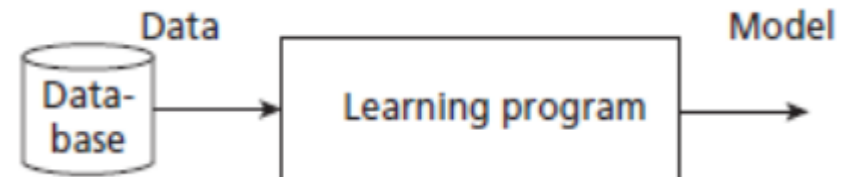
- **Machine Learning is the field of study that gives the computers ability to learn without being programmed.[Arthur Samuel]**
  - Here the input Data is used to **develop intelligent models**. The models will be used to predict new inputs.
  - The aim of ML is **to learn a model or set of rules** from the given data set automatically so that it can predict the unknown data correctly.

# Learning systems

For Humans



For Machine Learning



# What is a Model?

A Model can be any one of the following:

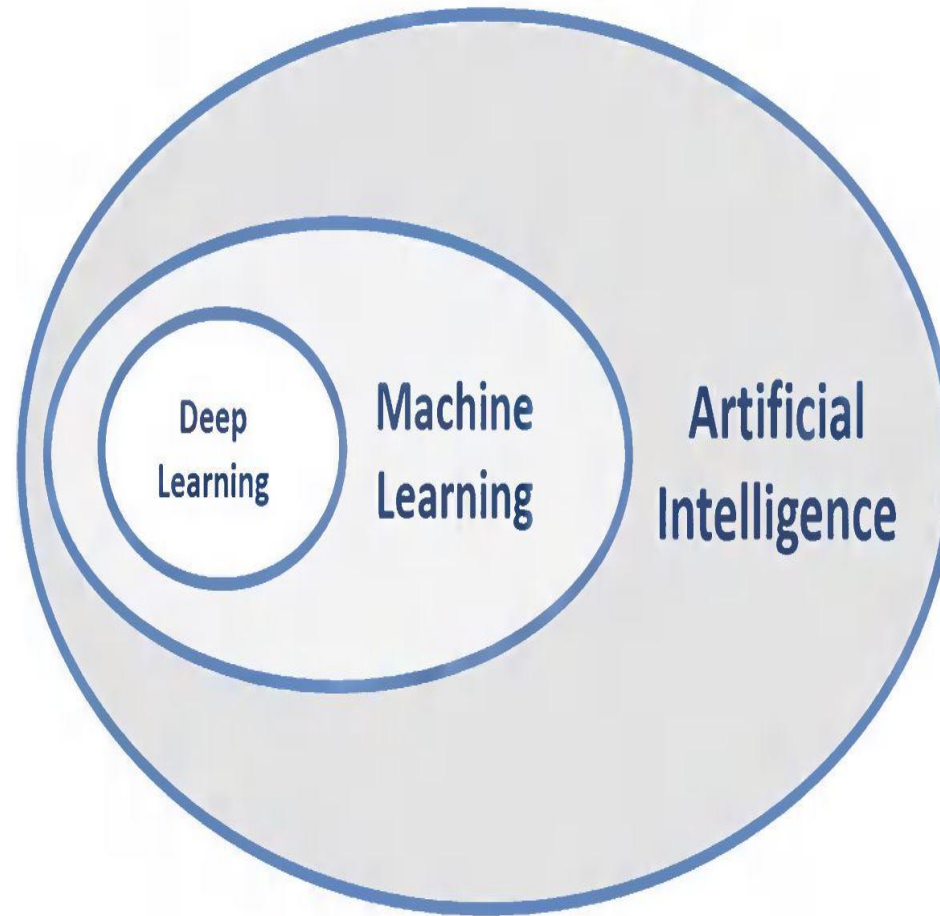
1. Mathematical Equation
2. Relational diagrams like Graphs/Trees
3. Logical if/else rules
4. Groupings called clusters

# Tom Mitchell definition of Machine learning

- Tom Mitchell, a computer scientist and professor at Carnegie Mellon University, provided a widely cited and influential definition of machine learning in his book titled "Machine Learning" (1997). The definition is as follows:
- ***"A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$  if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ ."***

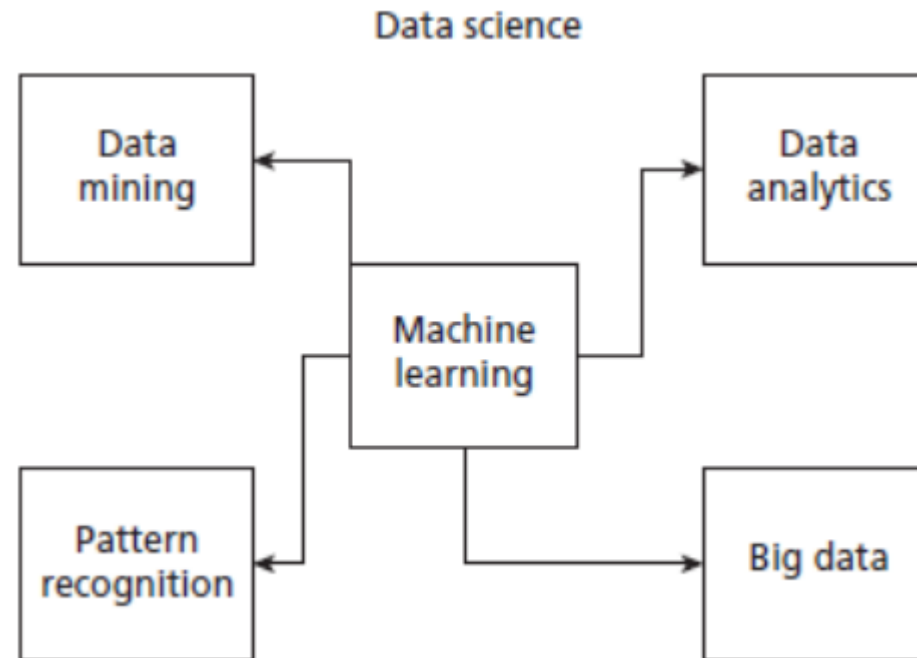
# Relationship of ML with Other Fields

# Relationship of AI with ML



# Machine Learning and Data Science

Data science is an “umbrella term” covering from data collection to data analysis.



# CHARACTERISTICS OF BIG DATA

- 1. Volume:** Big Data involves **massive amounts of data** that can be in petabytes or exabytes.
- 2. Velocity:** Describes the **speed at which data is generated**, collected, and processed.
- 3. Variety:** Encompasses the **different types of data sources** and formats. Big Data often includes structured data (e.g., databases), unstructured data (e.g., text, images, videos), and semi-structured data (e.g., JSON or XML files).
- 4. Veracity:** Refers to the **quality and reliability** of the data. Big Data sources can be messy and may include inaccuracies, inconsistencies, and errors.
- 5. Value:** Focuses on the importance of **turning data into value**. The ultimate goal of working with **Big Data is to extract meaningful insights and value from the massive amounts of data collected**

# Data Science and Data Mining

- **Data Science** is a multidisciplinary field that uses scientific methods, processes, algorithms, and systems to **extract knowledge and insights** from structured and unstructured data.
- **Data Mining** is a **specific step** within the broader field of data science. It involves the process of *discovering patterns, relationships, and insights from large datasets using various techniques, including statistical analysis, machine learning, and artificial intelligence.*

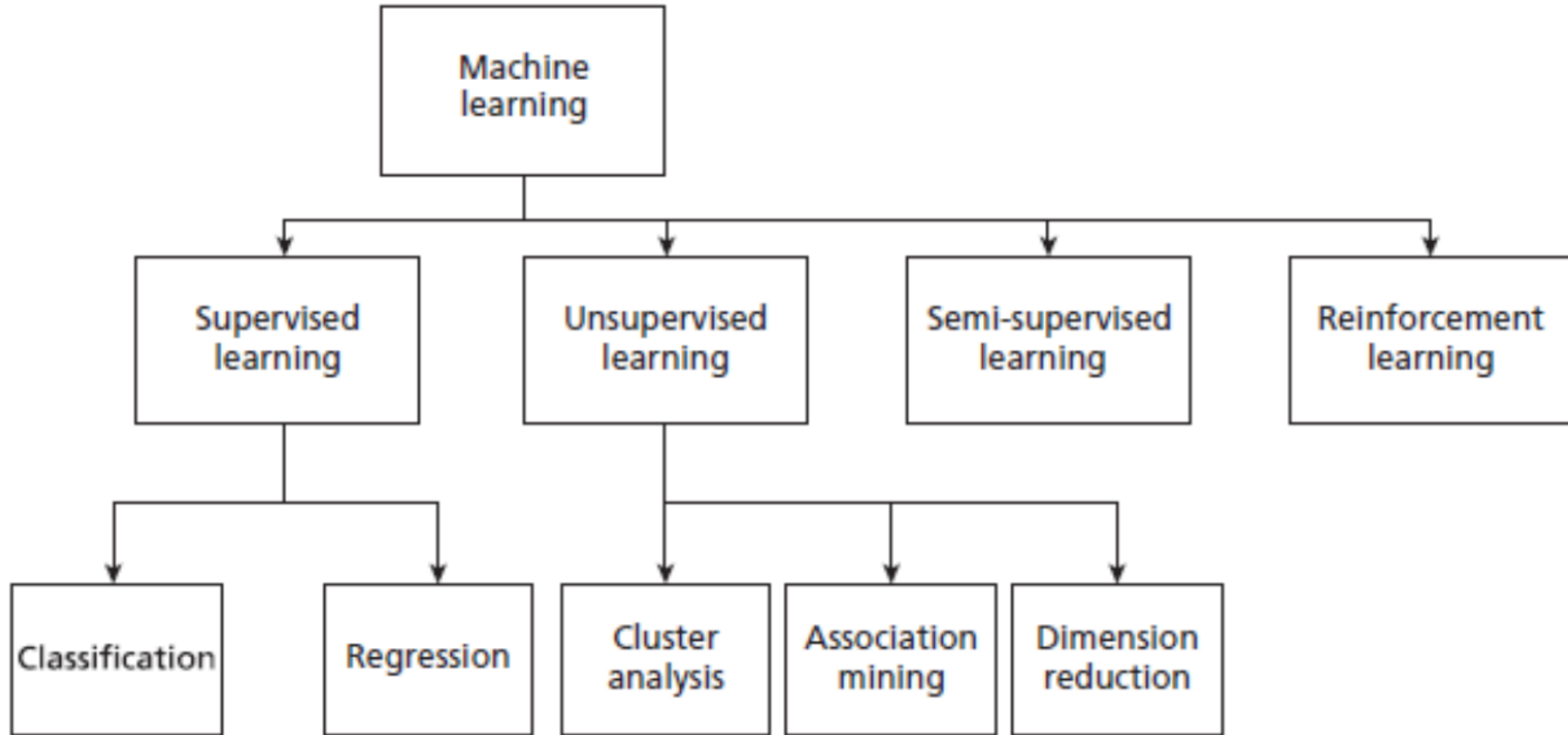
# Data science and Data analytics / Pattern recognition

- **Data Science, Data Analytics, and Pattern Recognition** are closely related fields, and they share **common goals of extracting insights and knowledge** from data. However, each field has its own focus and methodologies.
- **Data Analytics** is the process of **examining, cleaning, transforming, and modeling data** to discover useful information, draw conclusions, and support decision-making.
- **Pattern Recognition** is the process of **automatically recognizing patterns** in data and making decisions based on those patterns.

# Machine Learning and Statistics

- **Machine Learning** is a subfield of artificial intelligence (AI) that **focuses on the development of algorithms and models** that enable computers to learn from data and make predictions or decisions without explicit programming.
- **Statistics** is a branch of mathematics that involves **collecting, analyzing, interpreting, presenting, and organizing data**. It provides methods for making inferences from data, drawing conclusions, and quantifying uncertainty.

# Machine Learning types



# Labelled and Unlabeled Data



- Data is a **raw fact**.
- Data is represented in the form of **data table**
- Data also can be referred to as a **data point, sample or an example**
- Features are **attributes or characteristics** of an object. **Columns** of table are attributes.
- The most important attribute is **LABEL**. Label is the feature that we **aim to predict**.
- There are two types of data : **Labelled Data and Unlabeled Data**

# Labelled Data

**Table 1.1: Iris Flower Dataset**

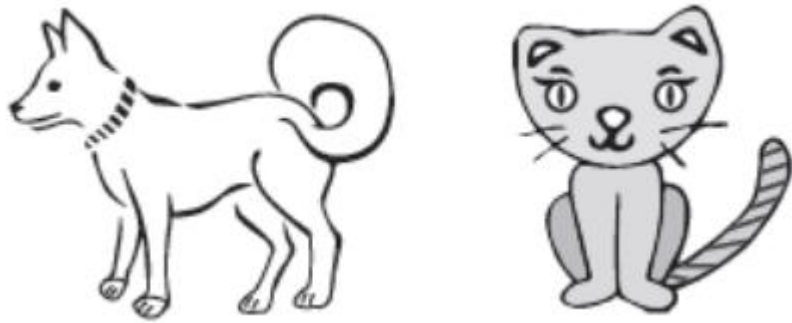
S.No.	Length of Petal	Width of Petal	Length of Sepal	Width of Sepal	Class
1.	5.5	4.2	1.4	0.2	Setosa
2.	7	3.2	4.7	1.4	Versicolor
3.	7.3	2.9	6.3	1.8	Virginica

A dataset need not be always numbers. It can be images or video frames. Deep neural networks can handle images with labels. In the following Figure 1.6, the deep neural network takes images of dogs and cats with labels for classification.

Input	Label
	dog
	Cat

# Unlabelled Data

DATA THAT IS NOT ASSOCIATED WITH LABELS ARE CALLED UNLABELLED DATA



## Labeled Data



Cat



Cat



Dog



Dog

## Unlabeled Data



# Supervised Learning

- **Supervised learning** is a type of machine learning where the algorithm is trained on a **labeled dataset**, meaning that the **input data used for training is paired with corresponding output labels**.
- The goal of supervised learning is **to learn a mapping from input features to the target output by generalizing patterns from the labeled training data**.
- Once trained, the model can **make predictions or classifications on new, unseen data**.

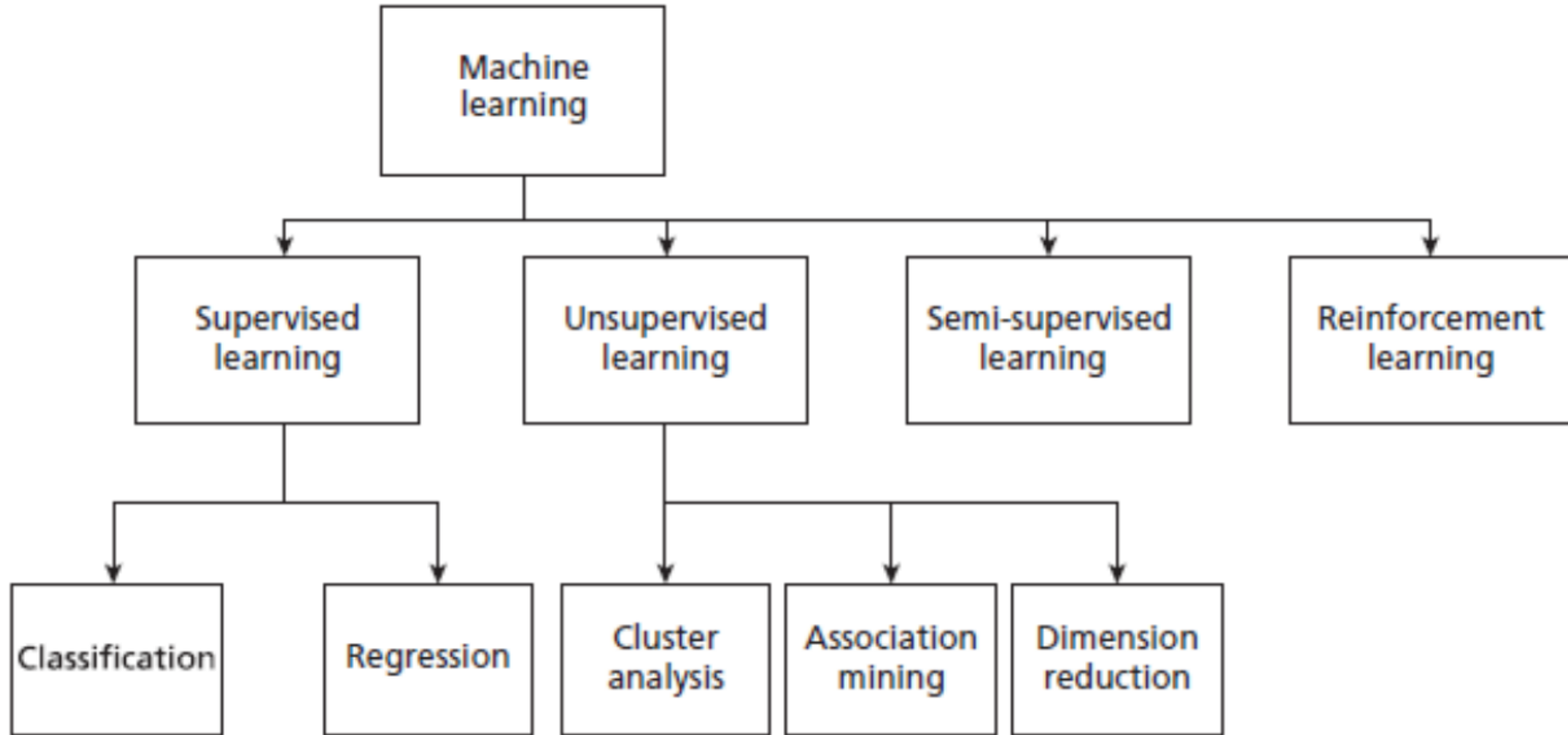
# Key Components of SL

- **Input Features (X)**
- **Output Labels (Y)**
- **Training Data**
- **Model**
- **Testing Data**

# Process of Supervised Learning

- 1.Data Collection:** Gather a labeled dataset that includes input features and corresponding output labels.
- 2.Data Preprocessing:** Clean and preprocess the data, handling missing values, scaling features, and encoding categorical variables.
- 3.Model Selection:** Choose an appropriate supervised learning algorithm based on the problem type (regression or classification) and characteristics of the data.
- 4.Training:** Feed the labeled data into the chosen model, allowing it to learn the patterns and relationships between input features and output labels.
- 5.Evaluation:** Assess the performance of the trained model on a separate validation or test dataset using metrics relevant to the specific task (e.g., mean squared error for regression or accuracy for classification).
- 6.Prediction:** Deploy the trained model to make predictions or classifications on new, unseen data.

# Machine Learning types



# Types of Supervised Learning:

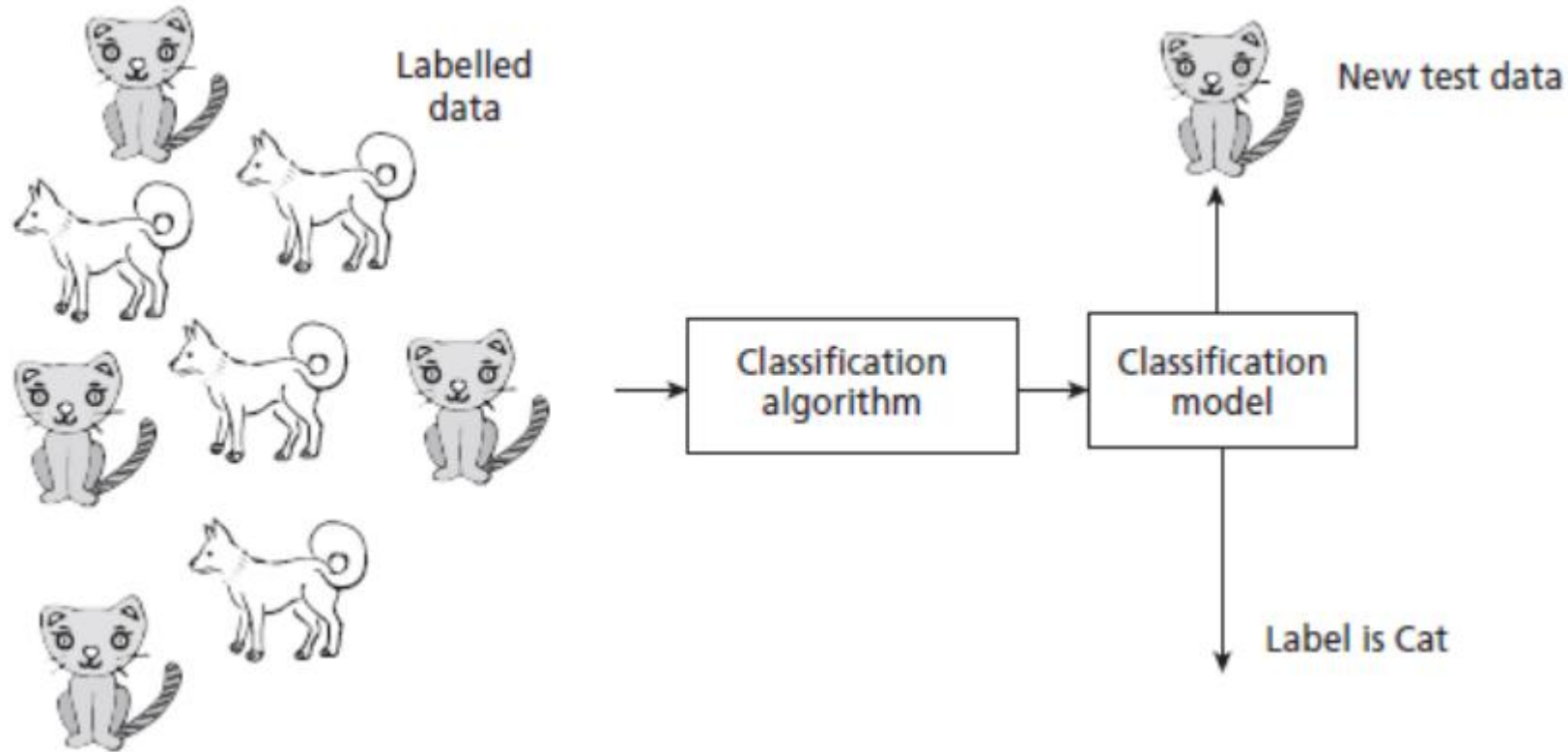
**1. Classification:** In classification tasks, the goal is to assign input data to **specific categories or classes**.

- Examples include *spam detection, image recognition, or sentiment analysis*.

**2. Regression:** In regression tasks, the goal is to predict a **continuous output variable**.

- Examples include predicting **house prices, temperature, or stock prices**.

# 1. Classification



# Algorithms : Classification

1	<b>Logistic Regression</b>
2	<b>Decision Trees</b>
3	<b>Random Forest:</b>
4	<b>Support Vector Machines (SVM)</b>
5	<b>K-Nearest Neighbors (KNN)</b>
6	<b>Naive Bayes</b>
7	<b>Neural Networks (Deep Learning)</b>

## 2. Regression

- Regression algorithms are used in machine learning to model and predict **continuous numeric values**.

# 2. Regression

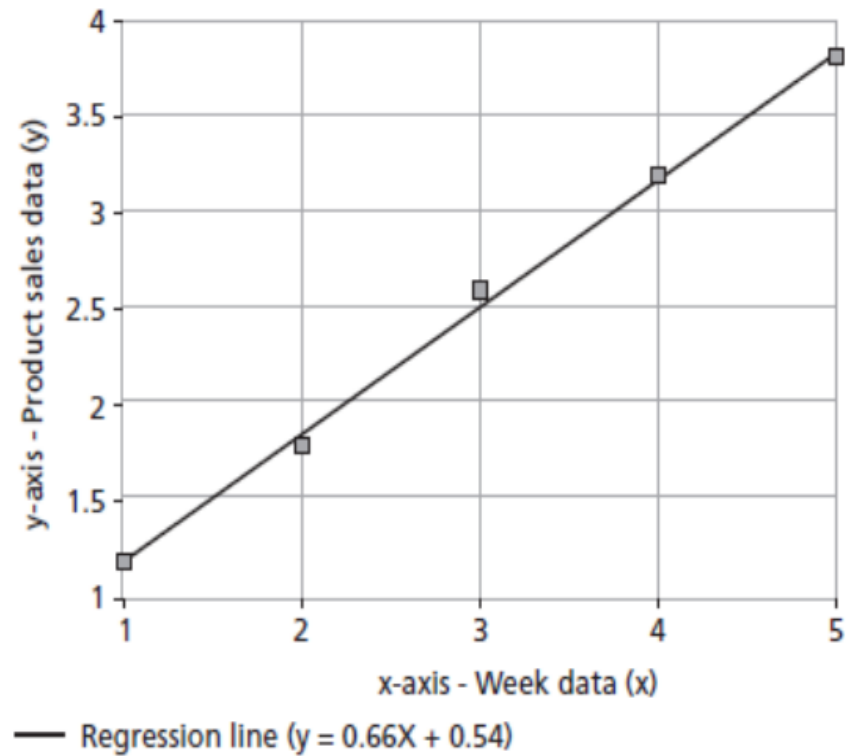
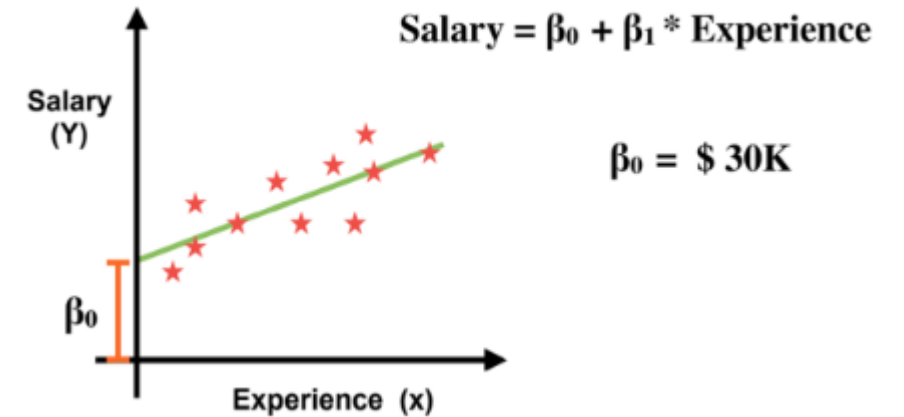


Figure 1.8: A Regression Model of the Form  $y = ax + b$

$$\text{product sales} = 0.66 \times \text{Week} + 0.54.$$



# Algorithms : Regression

1	<b>Linear Regression</b>
2	<b>Decision Trees</b>
3	<b>Random Forest</b>
4	<b>Support Vector Regression (SVR)</b>
5	<b>K-Nearest Neighbors (KNN) Regression</b>

# Unsupervised Learning

- **Unsupervised learning** is a type of machine learning where the algorithm is given data without explicit instructions on what to do with it.
- The system tries to learn the **patterns and the structure** from the data without labeled responses to guide the learning process.
- Unsupervised learning is often used for **exploratory data analysis** and pattern discovery.
- The two main types of unsupervised learning are **clustering and dimensionality reduction**.

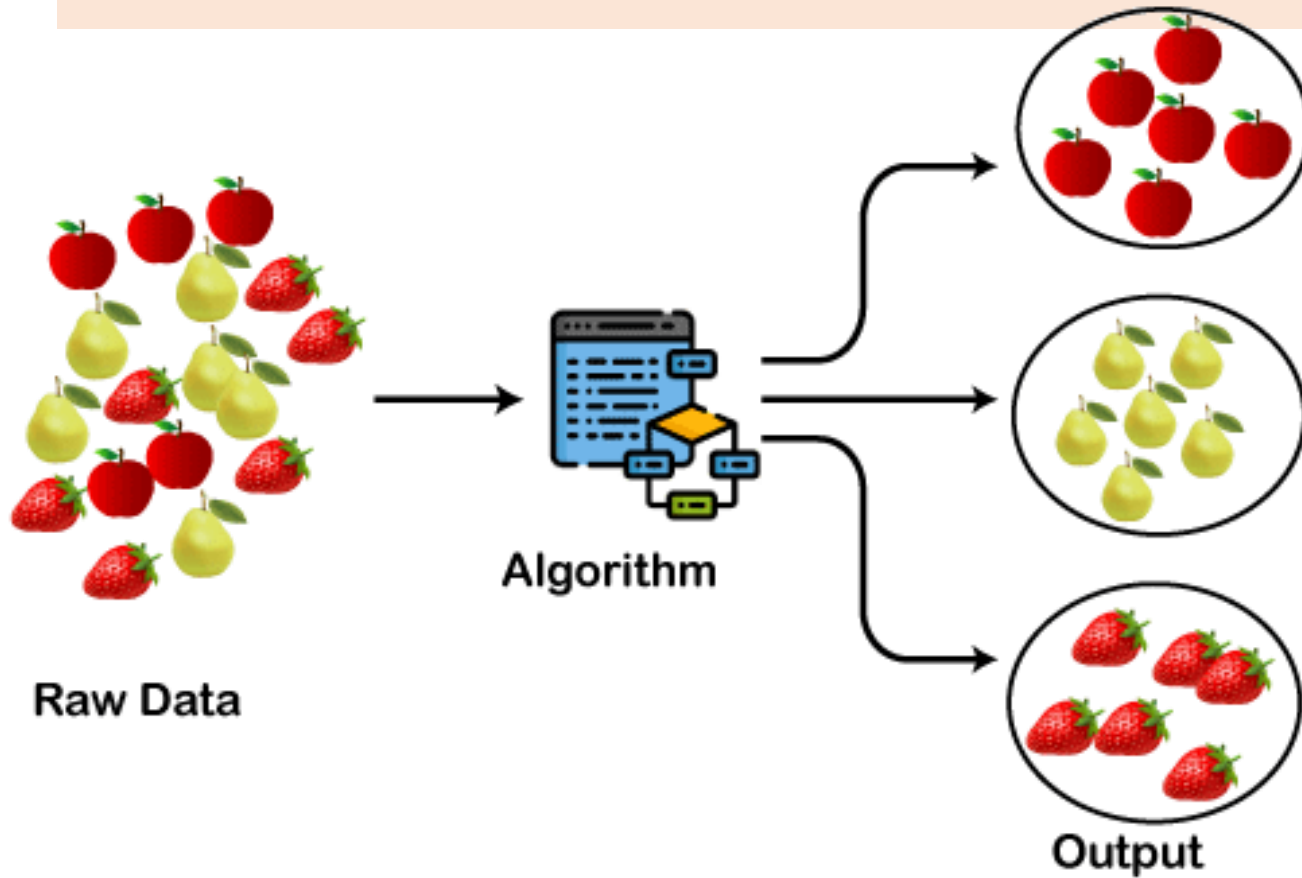
# Unsupervised Learning

- **Unsupervised learning** is particularly useful in scenarios where **labeled data is scarce or unavailable**.
- It allows the algorithm to discover **hidden patterns and structures** in the data without explicit guidance.
- The choice of algorithm depends on the **specific task, characteristics of the data, and the goals of the analysis**.

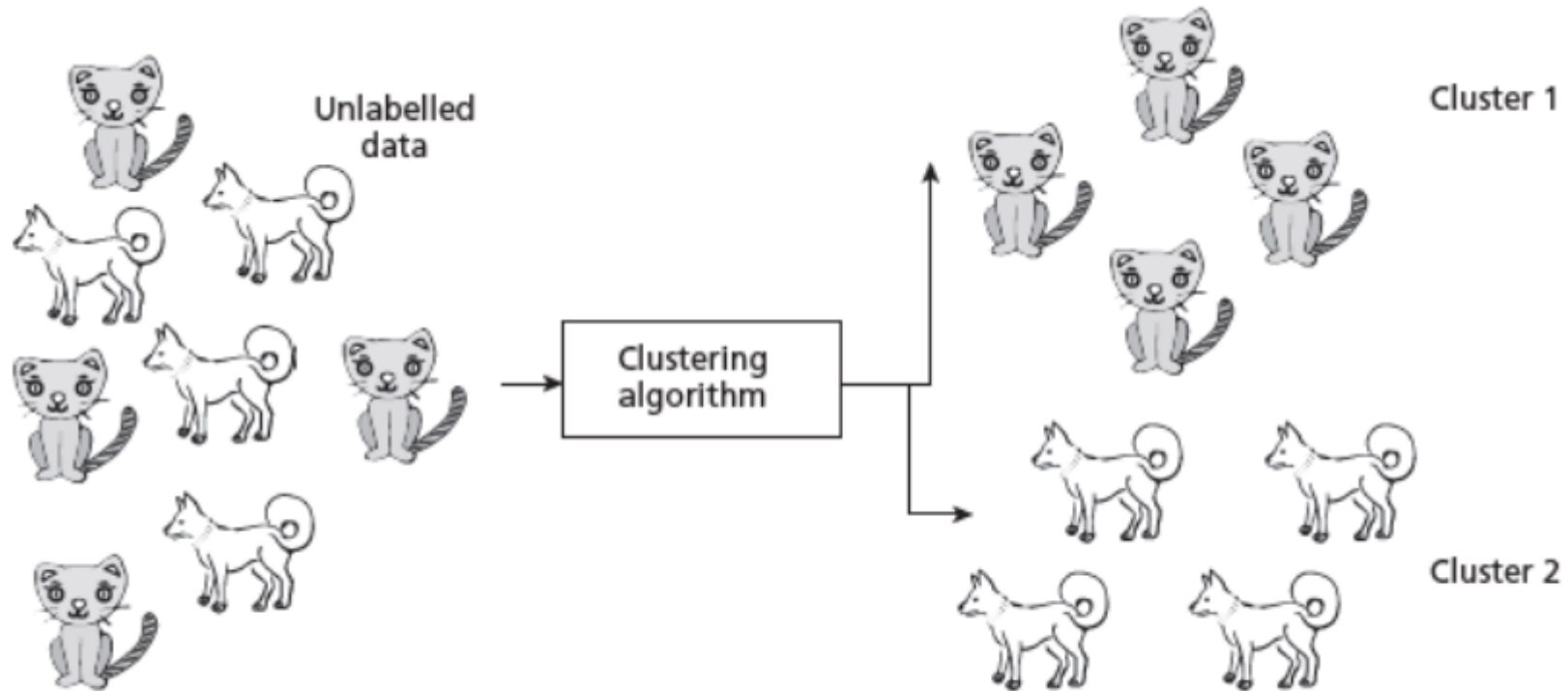
# Types of Unsupervised Learning

- 1. Clustering**
- 2. Dimensionality Reduction**
- 3. Association Rule Mining**

# Clustering



# Clustering



**Figure 1.9: An Example Clustering Scheme**

# Algorithms : Clustering

1	<b>K-Means Clustering</b>
2	<b>Hierarchical Clustering</b>

# Algorithms : Dimensionality Reduction

1	<b>Principal Component Analysis (PCA)</b>
2	<b>Autoencoders</b>
3	<b>Independent Component Analysis (ICA)</b>

## Supervised learning

Input data is labelled

There is a training phase

Data is modelled based on training dataset

Divided into two types:

Classification and Regression

Known number of classes (for classification)

## Unsupervised learning

Input data is unlabelled

There is no training phase

Uses properties of given data for classification

Most popular types: Clustering and

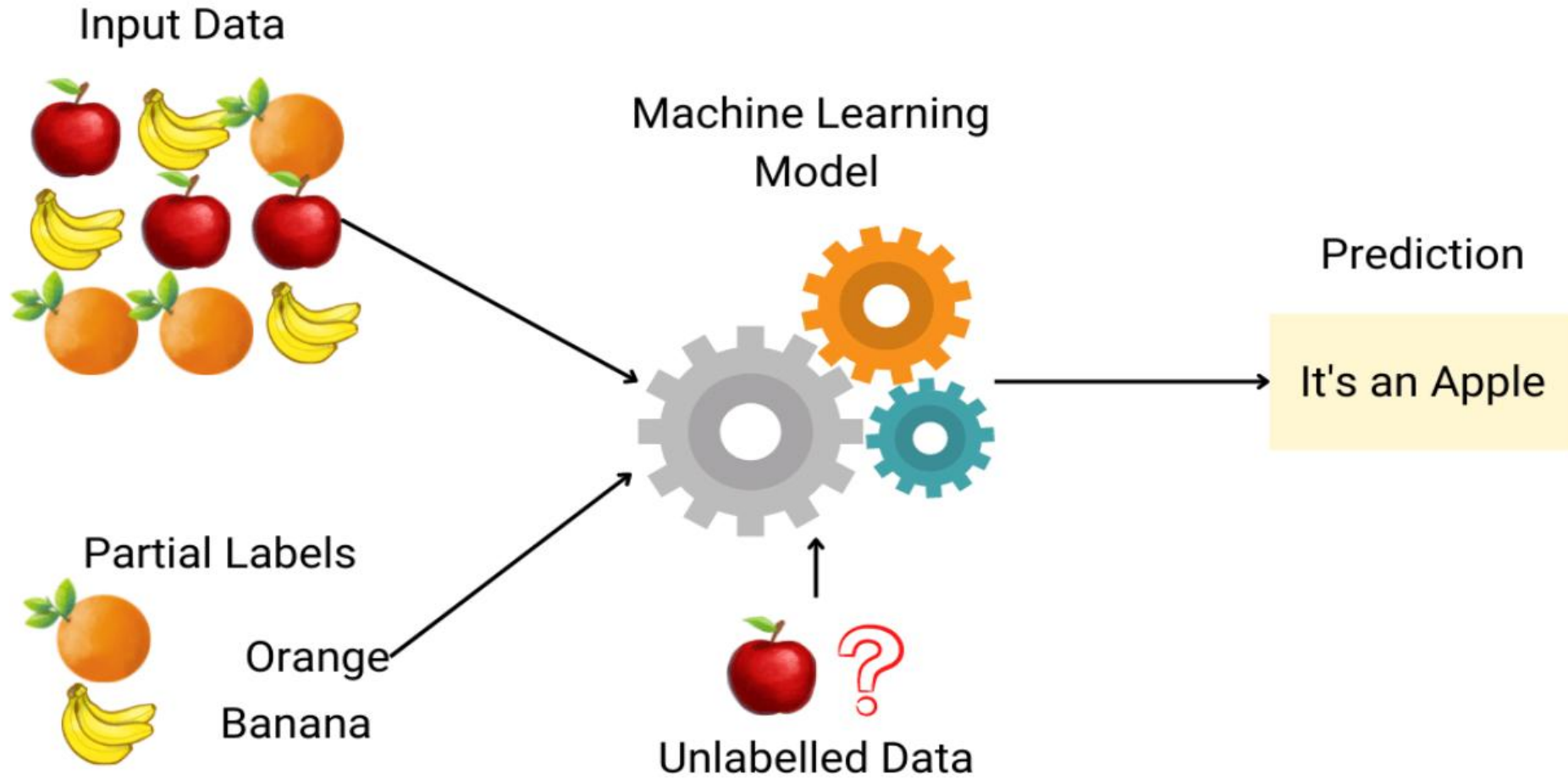
Dimensionality reduction

Unknown number of classes

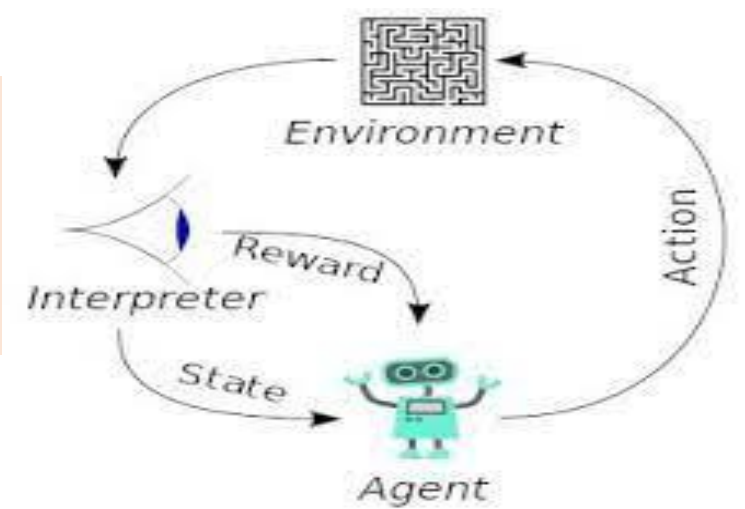
# Semi Supervised Learning

- Semi-supervised learning is a type of machine learning where the algorithm is **trained on a dataset that contains both labeled and unlabeled data.**
- In other words, only a subset of the training data has **explicit labels**, while the majority of the data **is unlabeled.**
- Semi-supervised learning aims to leverage the information from both labeled and unlabeled data to **improve the performance of the model.**

# Semi Supervised Learning

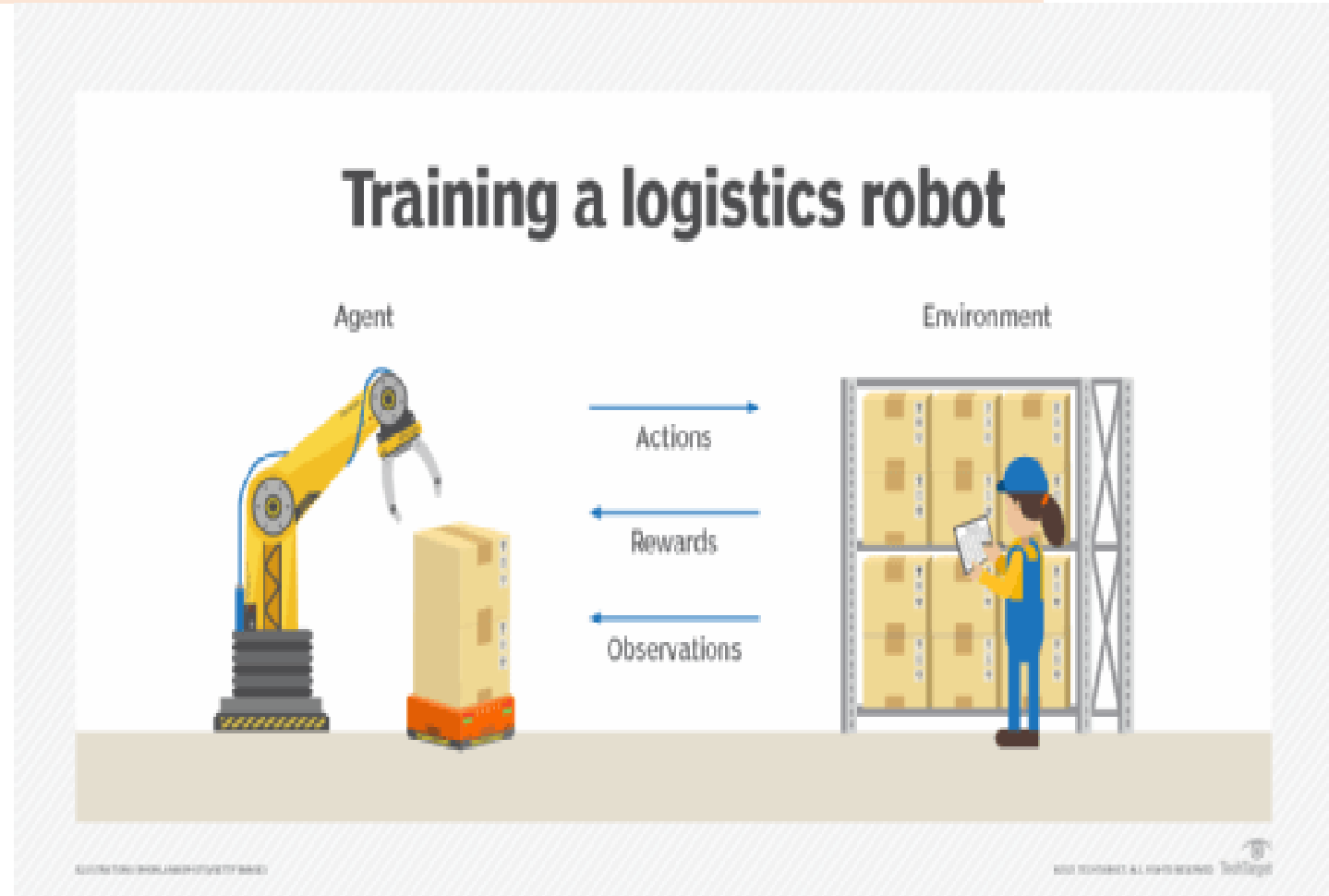
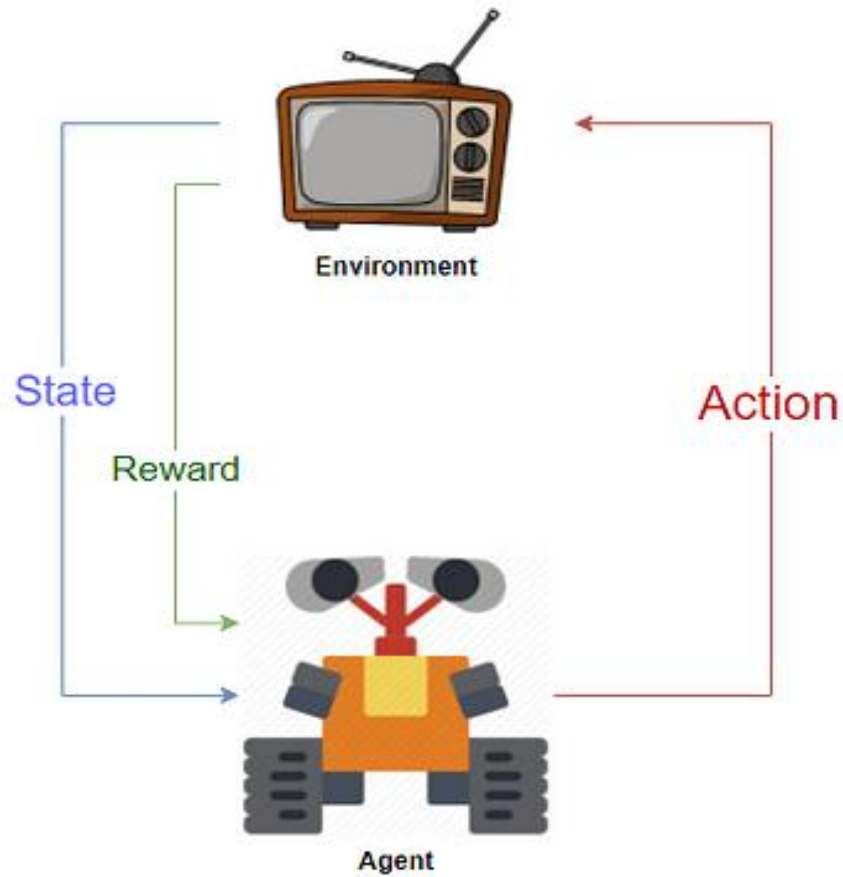


# Reinforcement Learning



- Reinforcement Learning (RL) is a **type of machine learning paradigm** where an agent learns to make decisions by interacting with an environment.
- The agent takes **actions**, **receives feedback in the form of rewards or penalties**, and learns to optimize its behavior over time to achieve a specific goal.
- In RL, **the agent is not** explicitly told which actions to take but discovers the optimal strategy through trial and error.

# Reinforcement Learning



# Reinforcement Learning

Block

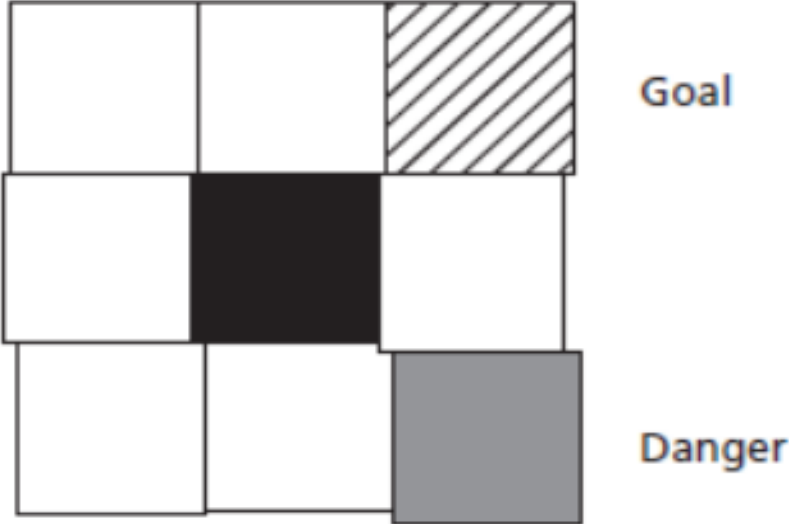


Figure 1.10: A Grid game

agent



environment



# Challenges of Machine Learning

1. ILL-POSED PROBLEMS – PROBLEMS WHOSE SPECIFICATIONS ARE NOT CLEAR
2. HUGE DATA
3. HUGE COMPUTATION POWER
4. COMPLEXITY OF ALGORITHMS
5. BIAS-VARIANCE

# Example for Ill Posed Problem

Input ( $x_1, x_2$ )	Output ( $y$ )
1, 1	1
2, 1	2
3, 1	3
4, 1	4
5, 1	5

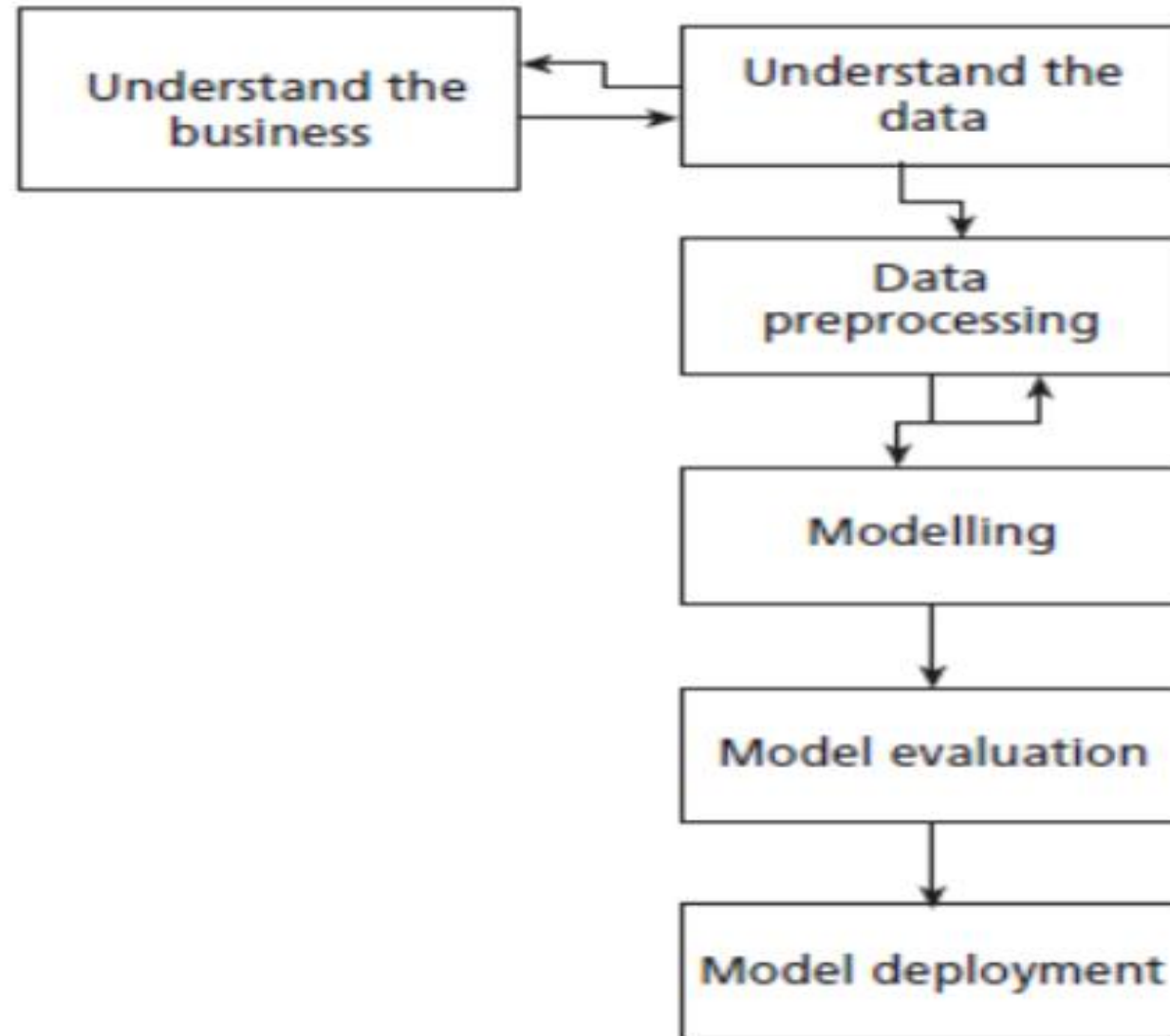
$$Y = x_1$$

$$Y = x_1/x_2$$

$$Y = x_1 * x_2$$

$$Y = x_1^x_2$$

# Machine Learning Process



# Machine Learning Applications

1. **Sentiment Analysis**
2. **Recommendation Systems**
3. **Voice Assistants**
4. **Google Maps**
5. **Facial Recognition**
6. **Object Detection**
7. **NLP**
8. **Financial Fraud Detection**
9. **Energy Management**
10. **E-commerce**
11. **Gaming**

# Machine Learning Applications

Sl.No	Problem Domain	Applications
1	Business	Predicting the bankruptcy of a business firm
2	Banking	Prediction of bank loan defaulters and detecting credit card frauds
3	Image Processing	Image search engines, object identification, image classification and generating synthetic images
4	Audio/ Voice	Chatbots like Alexa, Microsoft Cortana. Developing chatbots for customer support, speech to text and text to voice
5	Telecommunication	Trend analysis and identification of bogus calls, fraudulent calls and its callers.

# Machine Learning Applications

Sl.No	Problem Domain	Applications
6	Games	Game programs for Chess, GO and Video games
7	Natural Language Translation	Google Translation, Text Summarization and sentiment analysis
8	Web Analysis and Services	Identification of Access patterns, detection of e-mail spams, viruses, personalized web services, search engines like Google, detection of promotion of user websites
9	Medicine	Prediction of disease
10	Multimedia and Security	Face Recognition/Identification biometric projects like identification of a person from a large image or video database.

# Module1 : Part2 : Understanding Data-1

1. Introduction,
2. Big Data Analysis Framework,
3. Descriptive Statistics,
4. Univariate Data Analysis and Visualization.

# What is Data?

- All Facts are data.
- In Computer Systems bits encode facts present in **numbers, text, images, audio and video**.
- Data can be human **interpretable** (such as numbers or texts) or **diffused data** such as images or video that can be interpreted only by a computer.

# Amounts of Data

- Today data is growing in the order of **gigabytes, terabytes , exabytes.**
- **Byte:** 8bits,
- **Bit:** 0 /1
- **1KB:** 1024Bytes
- **1MB:** 1000KB
- **1GB:** 1,000,000 KB
- **1TB :** 1000 GB
- **1EB :** 1,000,000TB

# Types of Data (Big Data)

Big Data consists of three main types of data:

- 1. Structured Data,**
- 2. Unstructured data, and**
- 3. Semi-Structured Data.**

# Structured Data

- Structured data is stored in an **organized manner**, typically in databases, where it can be accessed in **tabular form**.
- This data can be easily retrieved using tools like **SQL**.
- Some **common types of structured data** in machine learning include:
  1. **Record Data**
  2. **Data Matrix**
  3. **Graph Data**
  4. **Ordered Data**

# 1. Record Data

- A collection of measurements from a process, where each object in the dataset has a **set of attributes**.
- It is often stored in a matrix format, where rows represent objects (**entities, cases, or records**), and columns represent attributes (**features or fields**).
- Labels describe **individual observations** in the dataset.

## 2.Data Matrix

- A variation of **record data** containing numeric attributes.
- **Standard matrix** operations can be performed.
- Data is represented **as points or vectors** in multidimensional space, where **each attribute serves as a dimension**.

# 3. Graph Data

- Represents **relationships** among objects.
- **Example:** A **webpage linking to another webpage** can be modeled as a graph, where webpages are **nodes** and hyperlinks are **edges**.

# 4. Ordered Data

- Data objects have an inherent order

## Examples of ordered data:

- **Temporal Data** – Data associated with time (e.g., customer purchase patterns during festivals). Time series data is a special type of sequential data where measurements occur over time.
- **Sequence Data** – Similar to sequential data but without timestamps (e.g., DNA sequences containing A, T, G, and C).
- **Spatial Data** – Contains attributes like positions or areas (e.g., maps where points relate to locations).

# Unstructured Data

- Unstructured data includes various formats such as **video, images, audio, textual documents, programs, and blog entries**.
- It is estimated that **80% of all data** falls under the unstructured category.

# Semi-Structured Data

- Semi-structured data contains elements of **both structured and unstructured data**.
- Examples include **XML/JSON data, RSS feeds, and hierarchical data**.

# Data Storage and Representation

- Once data is gathered, it must be **stored in a structure suitable** for analysis.
- **Efficient data storage and management** ensure accessibility for analytical processes.
- Different approaches to organizing and managing data exist, from flat files to databases.
  1. Flat Files
  2. Database Systems
  3. Spatial Databases

# Flat Files

- Flat files are **the simplest and most widely used** data sources. They are also the least expensive way to organize data.
- These files store data in **plain ASCII or EBCDIC formats**.
- Minor formatting differences can impact data mining algorithms, making flat files **suitable for small datasets** but **inefficient for handling large-scale data**.

# Some common types of flat files:

- **CSV files (Comma-Separated Values)** – Store values separated by commas. They are widely used in spreadsheet applications.
- **TSV files (Tab-Separated Values)** – Store values separated by tabs.

Both **CSV** and **TSV** files are widely used due to their generic format and compatibility with tools like **Google Sheets** and **Microsoft Excel**.

# Database Systems

- Databases store and manage structured data efficiently through **database management systems (DBMS)**.
- DBMS improves performance by including functionalities like **administration, query processing, and transaction management**.
- A **relational database** consists of tables, where:
  1. **Rows** represent records (or tuples).
  2. **Columns** represent attributes (or fields).
  3. Users can **manipulate and retrieve data using SQL**.

# Types of Databases:

- **Transactional Databases** – Contain transactional records with timestamps and identifiers. These databases are primarily used for associational analysis to detect correlations between items.
- **Time-Series Databases** – Store data indexed by timestamps, tracking sequential data such as sales trends, stock prices, and sensor readings.

# Spatial Databases

- Spatial databases contain **geographic information** stored in **raster or vector formats**.
- **Raster format** stores images (e.g., maps as pixel-based data).
- **Vector format** represents maps using **geometric shapes** (e.g., points, lines, and polygons)

# Other Data Representations

- **World Wide Web (WWW)** – A vast source of online information, used in data mining to extract valuable patterns.
- **XML (eXtensible Markup Language)** – A structured format for sharing data across platforms.
- **Data Streams** – Continuous data flow in real time, characterized by high volume and fixed order constraints.
- **RSS (Really Simple Syndication)** – A format for sharing news feeds across websites.
- **JSON (JavaScript Object Notation)** – A widely used format in machine learning algorithms.

# Big Data Analytics

- The primary goal of data analytics is to help businesses **make informed decisions**.
- For example, companies may use data analytics to determine **which products sell best** and optimize their marketing strategies.

# Difference Between Data Analysis and Data Analytics

- **Data Analysis** refers to **processing and interpreting historical data.**
- **Data Analytics** is a broader term that includes **data collection, preprocessing, and prediction.**

# Types of Data Analytics

- **Descriptive Analytics** – Summarizes the main characteristics of the data, focusing on what happened.
- **Diagnostic Analytics** – Investigates the cause-and-effect relationships to answer "Why did this happen?".
- **Predictive Analytics** – Uses historical data to forecast future trends.
- **Prescriptive Analytics** – Provides recommendations on what actions should be taken to optimize outcomes.

# Descriptive Analytics

Descriptive analytics summarizes data **after collection and cleaning**. It is often considered **basic statistical analysis** and is divided into:

- **Descriptive Statistics** – Focuses on summarizing data features.
- **Inferential Statistics** – Uses probability models to make predictions.

# Diagnostic Analytics

Diagnostic analytics answers the question "**Why?**"

- It examines **cause-and-effect relationships**.
- For example, if a product underperforms, diagnostic analytics identifies **the underlying reasons**.
- Various **factors and trends** are analyzed to understand what influenced past outcomes.

# Predictive Analytics

- Predictive analytics focuses on **forecasting future trends** based on historical data patterns. It answers the question: **"What will happen in the future?"**
- This process involves applying **algorithms and machine learning models** to analyze data and predict potential outcomes.
- Predictive analytics is a core component of **machine learning** and plays a significant role in decision-making.

# Prescriptive Analytics

- Prescriptive analytics goes a step further by determining the **best course of action** for a given scenario.
- It combines insights from **predictive analytics** with recommendations to mitigate risks.
- Helps organizations **optimize decision-making** and plan for future events.

# BIG DATA ANALYSIS FRAMEWORK

- Several frameworks exist for **big data analytics**, all of which follow a **layered architecture** to ensure efficiency and flexibility. A typical **4-layer architecture** consists of:

**1.Data Connection Layer**

**2.Data Management Layer**

**3.Data Analytics Layer**

**4.Presentation Layer**

# Data Connection Layer

- This layer is responsible for **data ingestion**, which involves collecting raw data and importing it into appropriate storage structures.
- It also performs **Extract, Transform, Load (ETL) operations** to prepare data for further processing.

# Data Management Layer

- This layer handles **data preprocessing and storage** while ensuring smooth execution of **queries and read/write operations**.

## It includes:

- **Data-in-place techniques**, where data is processed without movement.
- **Data repositories**, such as **data warehouses**, which store and retrieve data on demand.

# Data Analytics Layer

- This layer applies **machine learning algorithms, statistical models,** and other analytical methods to **extract insights from data.**
- It also includes **model validation** techniques to ensure accuracy and reliability.

# Presentation Layer

- The **Presentation Layer** consists of tools such as **dashboards, applications, and visualizations** that display results from analytical engines and machine learning models.

The **Big Data processing cycle** follows a structured workflow that includes:

**1.Data Collection**

**2.Data Preprocessing**

**3.Application of Machine Learning Algorithms**

**4.Interpretation and Visualization of Results**

- This process is **iterative** and ensures that data remains suitable for **data mining**. Since the foundation of machine learning lies in **data collection and preprocessing**, these aspects are covered in detail in this chapter.

## 2.3.1 Data Collection

- The **first step** in working with datasets is data collection. It is estimated that a **significant portion of time** in analytics projects is spent gathering **high-quality** data, as better data yields **better results**.
- A '**Good Dataset**' should have the following characteristics:
- **Timeliness** – The data should be up to date and **not obsolete**.
- **Relevancy** – Data should be **relevant** to the problem and free from bias.
- **Understandability** – Data should be **interpretable** and sufficient for the required analysis.

# Types of Data Sources

- Data sources are broadly classified into:

## 1. Open/Public Data Sources

1. Freely available without copyright restrictions.
2. Examples:
  1. **Digital libraries** – Contain vast amounts of text and document images.
  2. **Scientific datasets** – Large collections of genomic, biological, and experimental data.
  3. **Healthcare systems** – Databases for **patients, health insurance, and bioinformatics**.

## 2. Social Media Data

1. Data generated from platforms like **Twitter, Facebook, YouTube, and Instagram**.
2. Produces a **massive volume** of structured and unstructured data.

## 3. Multimodal Data

1. Involves multiple data formats such as **text, video, audio, or a combination** of these.
2. Examples:
  1. **Videos with subtitles** (text + audio)
  2. **Images with metadata** (image + text)

## 2.3.2 Data Preprocessing

- **Understanding ‘Dirty’ Data**

In real-world scenarios, **raw data is often messy** and contains errors. Dirty data may include:

- **Incomplete Data** – Missing values or attributes.
- **Outliers** – Unusual values that deviate from the dataset pattern.
- **Inconsistent Data** – Errors in formatting or incorrect values.
- **Inaccurate Data** – Incorrect or misleading data points.
- **Duplicate Data** – Repeated entries.

## 2.3.2 Data Preprocessing

To ensure **accurate machine learning models**, raw data must be **cleaned and preprocessed**. This process is known as **data wrangling** and includes:

- **Correcting human errors** (e.g., typos, missing values).
- **Standardizing data formats** (e.g., date, numeric values).
- **Handling missing and duplicate values** to prevent inconsistencies.

# Example: 'Bad' Data

Below is an example dataset illustrating errors found in real-world data:

Patient ID	Name	Age	Date of Birth (DoB)	Fever	Salary
1	John	21	-	Low	-1500
2	Andre	36	-	High	Yes
3	David	5	10/10/1980	Low	""
4	Raju	136	-	High	Yes

# Issues in the Dataset

- **Missing Data** – The Date of Birth (DoB) is missing for John, Andre, and Raju.
- **Inconsistent Data** – David's age is '5', while his DoB suggests 1980, indicating an inconsistency.
- **Incorrect Values** – Salary for John is '-1500', which is invalid (negative salary does not exist).
- **Outliers** – Raju's age is '136', which is highly unusual and could be a typographical error.

# Cleaning Data

- **Handle Missing Data** – Fill missing values using techniques like **mean imputation** or **removal** if data is too incomplete.
- **Correct Inconsistencies** – Standardize formats and validate records.
- **Detect Outliers** – Use statistical methods to flag and remove extreme values.

Proper **data preprocessing** ensures that **machine learning models** receive high-quality input, leading to **better predictions and insights**.

# Missing Data Analysis

- One of the key steps in data cleaning is missing data analysis.
- This process involves filling in missing values, reducing noise, and identifying outliers while correcting inconsistencies.
- Proper handling of missing data helps in avoiding model overfitting and improves the quality of data mining.

# Several techniques can be used to address missing data:

1. **Ignore the Tuple** – If a data entry has missing values, particularly the class label, it can be ignored. However, this method is ineffective when a large portion of the dataset has missing values.
2. **Manual Filling** – A domain expert analyzes the data tables and manually fills in missing values. This approach is accurate but impractical for large datasets due to its time-consuming nature.
3. **Using Global Constants** – Missing attributes can be replaced with a predefined value, such as 'Unknown' or 'Infinity'. However, this may sometimes lead to misleading results.
4. **Attribute Value Replacement** – Missing values can be filled using the attribute's mean, such as replacing a missing income value with the average income.
5. **Class-Based Mean Replacement** – The mean value of an attribute within the same class is used to fill in missing values, providing a more context-aware approach.
6. **Probable Value Estimation** – Advanced techniques like classification or decision trees can predict and fill in the missing values based on existing data trends.

# Several techniques can be used to address missing data:

1. While these methods can improve data completeness, they may introduce bias, **leading to potential errors.**
2. The difference between the estimated and actual values is known as an **error or bias.**

# Removal of Noisy or Outlier Data

- Noise in data refers to random errors or variances in measured values. One effective way to remove noise is **binning**, a technique that sorts and groups values into equal-frequency bins (also called buckets). Neighbor values within bins help smooth noisy data.

Binning is often used as a **discretization technique**, helping to manage noisy data efficiently. Common binning techniques include:

- **Smoothing by Means** – Replacing all values in a bin with their mean.
- **Smoothing by Medians** – Replacing all values in a bin with their median.
- **Smoothing by Bin Boundaries** – Replacing values in a bin with the closest boundary value (either maximum or minimum in the bin).

# Example 2.1

- **Given dataset:**

$S = \{12, 14, 19, 22, 24, 26, 28, 31, 34\}$

- Apply different binning techniques assuming a bin size of 3.

# Solution:

- Using the **equal-frequency binning method**, the data is distributed into bins of size 3 as follows:
- **Bin 1:** 12, 14, 19
- **Bin 2:** 22, 24, 26
- **Bin 3:** 28, 31, 32

# Smoothing by Means

- Each bin is replaced by its mean value:
- **Bin 1:** 15, 15, 15
- **Bin 2:** 24, 24, 24
- **Bin 3:** 30.3, 30.3, 30.3

# Smoothing by Bin Boundaries

- Each value is replaced by the nearest boundary value within its bin:
- **Bin 1:** 12, 12, 19
- **Bin 2:** 22, 22, 26
- **Bin 3:** 28, 32, 32

In the bin boundary method, the minimum and maximum values in each bin serve as fixed boundaries, while the middle values are adjusted to the nearest boundary. For example, in **Bin 1**, the value **14** is compared with boundaries **12** and **19** and is adjusted to **12** as it is the closest value. This process is repeated for all bins.

# Data Integration and Data Transformations

- Data integration involves combining data from multiple sources into a unified dataset. However, this process can introduce redundant data.
- The primary objective of data integration is to identify and eliminate redundancies.
- Data transformation techniques, such as normalization, are essential for enhancing the efficiency of data mining algorithms.
- Transforming data ensures it is in a suitable format for processing, acting as a preliminary step before analysis.
- One widely used transformation method is **normalization**, which scales attribute values to a specific range (e.g., 0 to 1). This helps improve the performance of data mining techniques, particularly in neural networks.

# Two common normalization methods include:

1. Min-Max Normalization
2. Z-Score Normalization

# Min-Max Normalization

- Min-Max normalization scales each variable  $V$  based on its difference from the minimum value, divided by the data range, and maps it to a new range (such as 0 to 1). This technique is frequently applied in neural networks.

The formula for Min-Max normalization is:

$$\text{min-max} = \frac{V - \text{min}}{\text{max} - \text{min}} \times (\text{new max} - \text{new min}) + \text{new min}$$

Here:

- **max - min** represents the data range.
- **min** and **max** are the minimum and maximum values of the original dataset.
- **new max** and **new min** define the target range (e.g., 0 to 1).
- This transformation ensures that all values are proportionally scaled to fit within the desired range.

# Example 2.2: Min-Max Normalization

## Problem Statement:

Consider the dataset  $V=\{88,90,92,94\}$  . Apply the **Min-Max normalization** method and map the values to a new range of **0 to 1**.

## Solution:

- The minimum value in the set is **88**, and the maximum value is **94**.
- The new range is defined as **[0, 1]**, meaning **new min = 0** and **new max = 1**.

- Using the **Min-Max normalization** formula:

$$\text{min-max} = \frac{V - \text{min}}{\text{max} - \text{min}} \times (\text{new max} - \text{new min}) + \text{new min}$$

We calculate the normalized values as follows:

For **88**:

$$\frac{88 - 88}{94 - 88} \times (1 - 0) + 0 = \frac{0}{6} = 0$$

For **90**:

$$\frac{90 - 88}{94 - 88} \times (1 - 0) + 0 = \frac{2}{6} = 0.33$$

For 92:

$$\frac{92 - 88}{94 - 88} \times (1 - 0) + 0 = \frac{4}{6} = 0.66$$

For 94:

$$\frac{94 - 88}{94 - 88} \times (1 - 0) + 0 = \frac{6}{6} = 1$$

Thus, after Min-Max normalization, the original values {88, 90, 92, 94} are mapped to {0, 0.33, 0.66, 1}, ensuring that the normalized data falls within the range 0 to 1.

# z-Score Normalization

- **Definition:** This method normalizes data by subtracting the mean from each value and dividing the result by the standard deviation of the dataset. The formula is:

$$V^* = \frac{V - \mu}{\sigma}$$

where:

- $V^*$  is the normalized value,
- $V$  is the original value,
- $\mu$  is the mean of the dataset,
- $\sigma$  is the standard deviation.

# Calculate standard deviation for $V=\{10,20,30\}$ .

To calculate the standard deviation ( $\sigma$ ) for the set  $V = \{10, 20, 30\}$ , follow these steps:

## Step 1: Calculate the Mean ( $\mu$ )

$$\mu = \frac{\sum X}{N} = \frac{10 + 20 + 30}{3} = \frac{60}{3} = 20$$

## Step 2: Compute Each Value's Squared Deviation from the Mean

$$(10 - 20)^2 = (-10)^2 = 100$$

$$(20 - 20)^2 = (0)^2 = 0$$

$$(30 - 20)^2 = (10)^2 = 100$$

# Calculate standard deviation for $V=\{10,20,30\}$ .

## Step 3: Find the Variance ( $\sigma^2$ )

For population standard deviation:

$$\sigma^2 = \frac{\sum(X - \mu)^2}{N} = \frac{100 + 0 + 100}{3} = \frac{200}{3} \approx 66.67$$

For sample standard deviation:

$$s^2 = \frac{\sum(X - \mu)^2}{N - 1} = \frac{200}{2} = 100$$

# Calculate standard deviation for $V=\{10,20,30\}$ .

## Step 4: Compute the Standard Deviation

For population standard deviation:

$$\sigma = \sqrt{66.67} \approx 8.16$$

For sample standard deviation:

$$s = \sqrt{100} = 10$$

## Final Answer

- Population Standard Deviation:  $\approx 8.16$
- Sample Standard Deviation: 10

# Example : z-Score Computation

- **Problem Statement:** Consider the dataset  $V=\{10,20,30\}$ . Convert these values to **z-scores**.

## **Solution:**

The mean ( $\mu$ ) of the dataset is **20**, and the standard deviation ( $\sigma$ ) is **10**.

## Example 2.3: z-Score Computation

### Problem Statement:

Consider the dataset  $V = \{10, 20, 30\}$ . Convert these values to **z-scores**.

### Solution:

The mean ( $\mu$ ) of the dataset is **20**, and the standard deviation ( $\sigma$ ) is **10**. Using the z-score formula:

For **10**:

$$z = \frac{10 - 20}{10} = \frac{-10}{10} = -1$$

For **20**:

$$z = \frac{20 - 20}{10} = \frac{0}{10} = 0$$

For **30**:

$$z = \frac{30 - 20}{10} = \frac{10}{10} = 1$$

Thus, the z-scores for the dataset  $\{10, 20, 30\}$  are  $\{-1, 0, 1\}$ , respectively.

# Importance of z-Scores

- z-Scores are commonly used for **outlier detection**.
- If a z-score is less than **-3** or **greater than +3**, the value is **likely an outlier**.
- A drawback of z-score normalization is its sensitivity to outliers, as it depends on the dataset's mean and standard deviation.

# Data Reduction

- **Definition:**

Data reduction minimizes dataset size while preserving essential information.

**Techniques include:**

- **Data Aggregation:** Combining similar data points.
- **Feature Selection:** Choosing the most relevant attributes.
- **Dimensionality Reduction:** Reducing the number of variables while retaining important patterns.

# Descriptive Statistics

- Descriptive statistics is a branch of statistics that focuses on **summarizing and describing data**.
- It does not go beyond the representation of data and does not concern itself with **machine learning algorithms or predictive modeling**.

# Descriptive Statistics

- **Data visualization** is an essential aspect of descriptive statistics, as it helps in analyzing and presenting data effectively.
- By using **descriptive analytics and data visualization techniques**, one can understand the nature of data, which is crucial for determining the suitable machine learning or data mining methods.
- This process is known as **Exploratory Data Analysis (EDA)**. EDA is aimed at preparing data for machine learning models by summarizing its key characteristics.

# Dataset and Data Types

- A dataset is a collection of data objects, which may include records, points, vectors, patterns, events, cases, samples, or observations. Each of these objects contains multiple attributes, also known as properties or characteristics of an entity.
- For example, consider the following dataset:

**Table: Sample Patient Data**

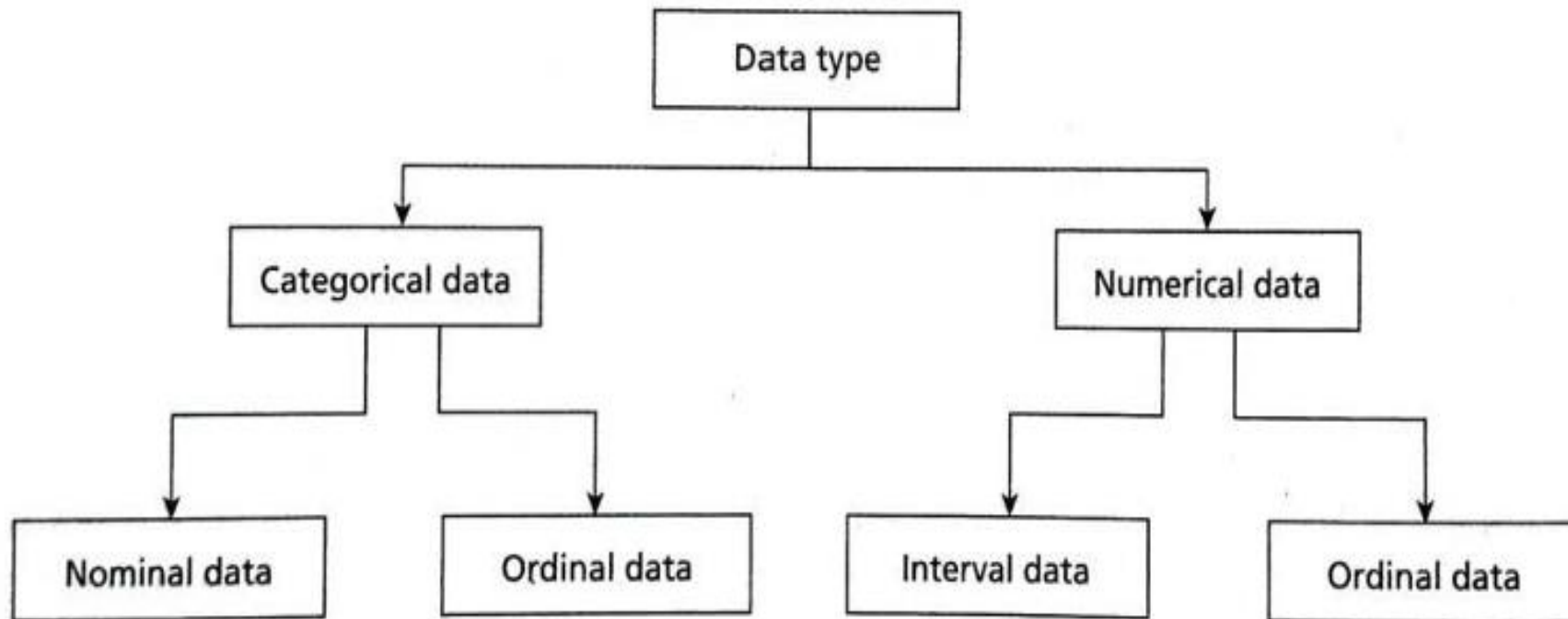
Patient ID	Name	Age	Blood Test	Fever	Disease
1	John	21	Negative	Low	No
2	Andre	36	Positive	High	Yes

- Each attribute is associated with a value, which is a part of the measurement process. The type of an attribute determines its **data type** or **measurement scale**.

# Types of Data

Broadly, data can be classified into two types:

- 1. Categorical or Qualitative Data**
- 2. Numerical or Quantitative Data**



# Categorical or Qualitative Data

- Categorical data can be further divided into two types:
- **Nominal Data:** Data that consists of labels or names with no inherent order.
- For example, **Patient ID** in the above table is nominal because it is just an identifier without any numerical significance.
- **Ordinal Data:** Data that has a meaningful order but does not have uniform differences between values.
- For example, **Fever levels (Low, Medium, High)** form an ordinal scale since there is an inherent ranking.

# Numerical or Quantitative Data

## Numerical data is divided into:

- **Interval Data:** Numeric data where the differences between values are meaningful, but there is no true zero point.
- Example: Temperature measured in Celsius or Fahrenheit.
- **Ratio Data:** Numeric data where both differences and ratios are meaningful, and there is a true zero point.
- Example: Age, weight, and height.

## Another classification of data is based on its nature:

- **Discrete Data:** Values that are counted as whole numbers, such as employee ID or number of patients in a hospital.
- **Continuous Data:** Values that can be measured and include decimals, such as age or weight.

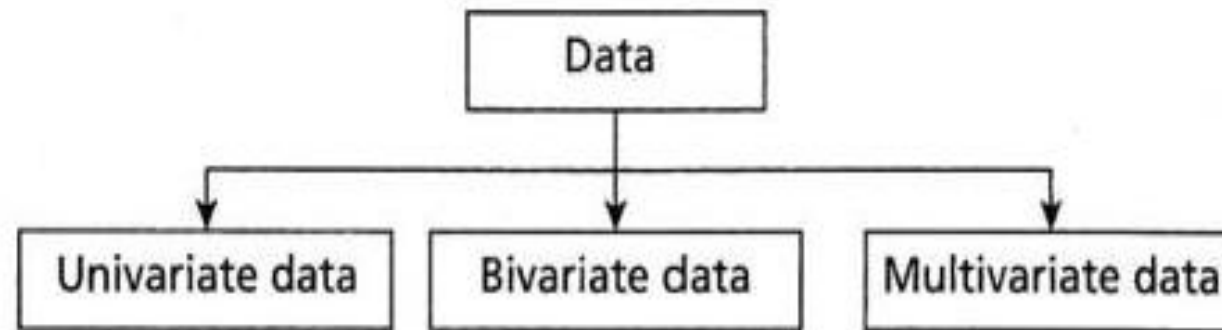
# Another classification of data is based on its nature:

- **Discrete Data:** Values that are counted as whole numbers, such as employee ID or number of patients in a hospital.
- **Continuous Data:** Values that can be measured and include decimals, such as age or weight.

# Classification Based on Variables

Data can also be classified based on the number of variables used in a dataset:

1. **Univariate Data:** Data with only one variable.
2. **Bivariate Data:** Data that involves two variables.
3. **Multivariate Data:** Data involving three or more variables.



# 1. Univariate Data

- Univariate data refers to data with a single variable or feature. The goal is typically to analyze or model the distribution, properties, or characteristics of that one variable.
- **Example 1: House Prices Prediction (Single Feature)**
  - **Variable:** Price of a house.
  - **Description:** You might have data on the prices of houses over time or across different regions, but no other features. The task is to understand the distribution of house prices.

# 1. Univariate Data

## Example 2: Customer Age Distribution

- **Variable:** Age of customers.
- **Description:** An analysis of customer age, where the dataset contains just the ages of customers. This can be used for clustering or basic statistics like mean, median, etc.

## Example 3: Stock Price of a Single Company

- **Variable:** Daily closing price of a stock.
- **Description:** You analyze only the stock price of a single company without considering other factors like trading volume or market indicators.

## 2. Bivariate Data

- Bivariate data refers to data with two variables, and the goal is often to examine the relationship between the two variables.
- **Example 1: Height vs. Weight**
  - **Variables:** Height and weight of individuals.
  - **Description:** The dataset contains measurements of people's heights and their corresponding weights. A common task could be to create a linear regression model to predict weight based on height.

## 2. Bivariate Data

### **Example 2: Temperature vs. Ice Cream Sales**

- **Variables:** Daily temperature and the amount of ice cream sold.
- **Description:** You might examine how the temperature on a given day affects the number of ice cream units sold, potentially using regression analysis to model the relationship.

### **Example 3: Income vs. Education Level**

- **Variables:** Income and level of education.
- **Description:** You can explore how an individual's income is related to their level of education, with the goal of identifying trends in the data.

# 3. Multivariate Data

- Multivariate data involves multiple variables or features. It's common in machine learning, as most real-world problems involve multiple factors influencing the outcome.
- **Example 1: House Prices Prediction (Multiple Features)**
  - **Variables:** Square footage, number of bedrooms, location, year built, etc.
  - **Description:** To predict the price of a house, you would use several features like the house's size, number of rooms, neighborhood, and other factors to train a machine learning model (e.g., linear regression, decision tree).

# 3. Multivariate Data

## Example 2: Customer Segmentation

- **Variables:** Age, income, location, purchasing behavior.
- **Description:** You use customer data with several features to identify different segments of customers (e.g., high-income, young customers who prefer tech products).

## Example 3: Medical Diagnosis (Multiple Symptoms and Test Results)

- **Variables:** Blood pressure, heart rate, cholesterol levels, age, weight, symptoms.
- **Description:** Predicting the likelihood of a certain medical condition based on a variety of health indicators and symptoms.

# Univariate Data Analysis and Visualization

- Univariate analysis is the most basic form of statistical analysis.
- It involves only one variable. This variable can be classified as a **category**.
- Univariate analysis does not **examine relationships or causation**; rather, its primary goal is to **describe data and identify patterns**.
- Describing univariate data involves ***analyzing frequency distributions, central tendency measures, dispersion or variability, and the overall shape of the data.***

# Data Visualization

- Graphical representation of data is essential for understanding and analyzing information. Data visualization enables better interpretation and communication of data insights. Some common graphs used in univariate data analysis include **bar charts, histograms, frequency polygons, and pie charts.**
- The key benefits of using graphs are:
  1. **Clear presentation of data**
  2. **Effective data summarization**
  3. **Easy description and exploration of data**
  4. **Enhanced data comparison capabilities**

# Bar Chart

- A bar chart (or bar graph) is commonly used to represent the frequency distribution of variables.
- It is particularly effective in displaying discrete data.
- Additionally, bar charts help illustrate nominal data and facilitate comparison of frequencies across different groups.
- For instance, consider the marks of students:  $\{45, 60, 60, 80, 85\}$  with Student IDs  $\{1, 2, 3, 4, 5\}$ . The corresponding bar chart is shown below:

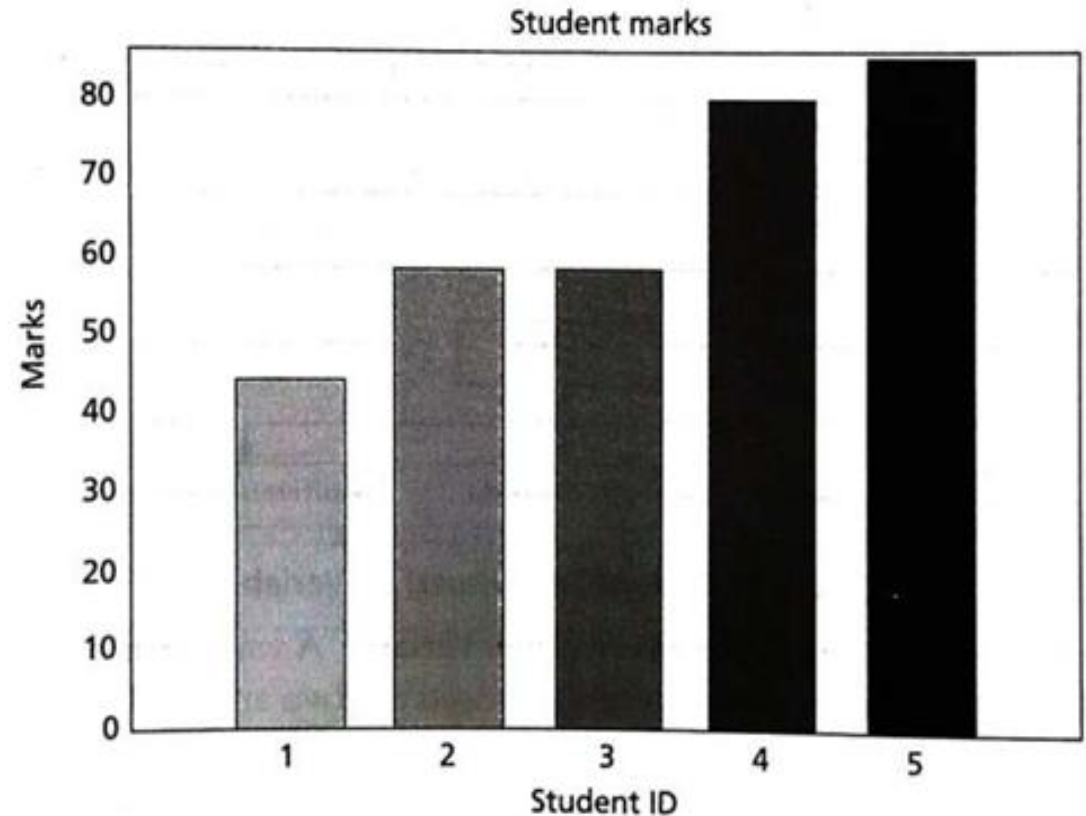


Figure 2.3: Bar Chart

## Bar Chart Example ▼

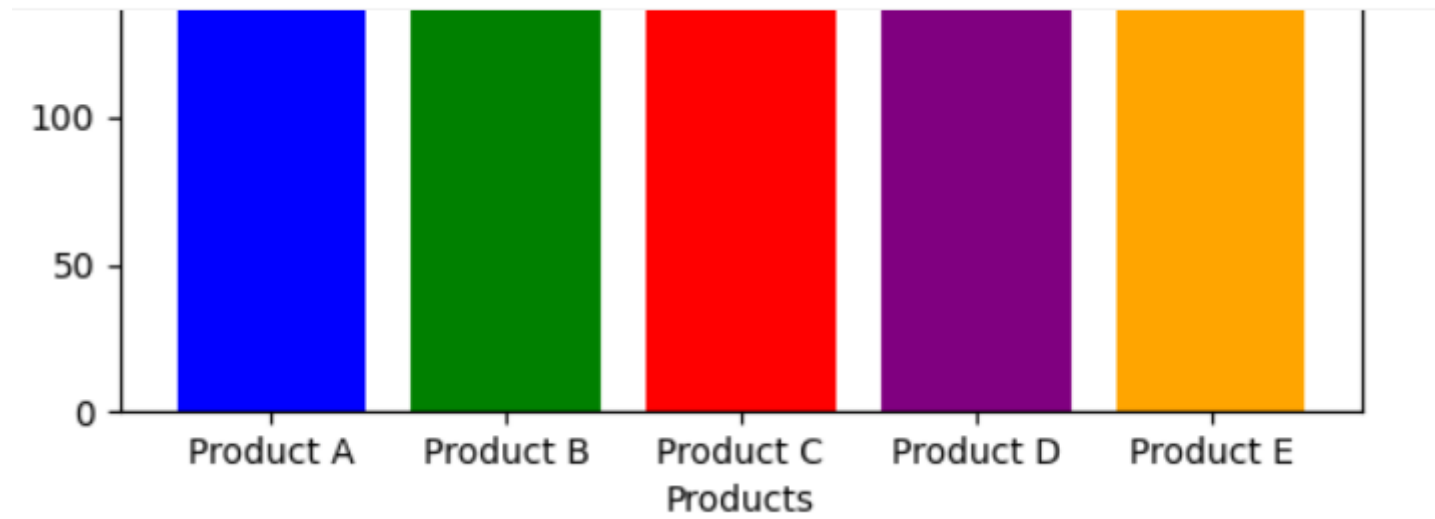
```
import matplotlib.pyplot as plt

# Sample dataset
products = ['Product A', 'Product B', 'Product C', 'Product D', 'Product E']
sales = [150, 200, 300, 250, 180]

# Creating the bar chart
plt.bar(products, sales, color=['blue', 'green', 'red', 'purple', 'orange'])

# Adding titles and labels
plt.xlabel('Products')
plt.ylabel('Sales')
plt.title('Sales of Different Products')

# Displaying the chart
plt.show()
```

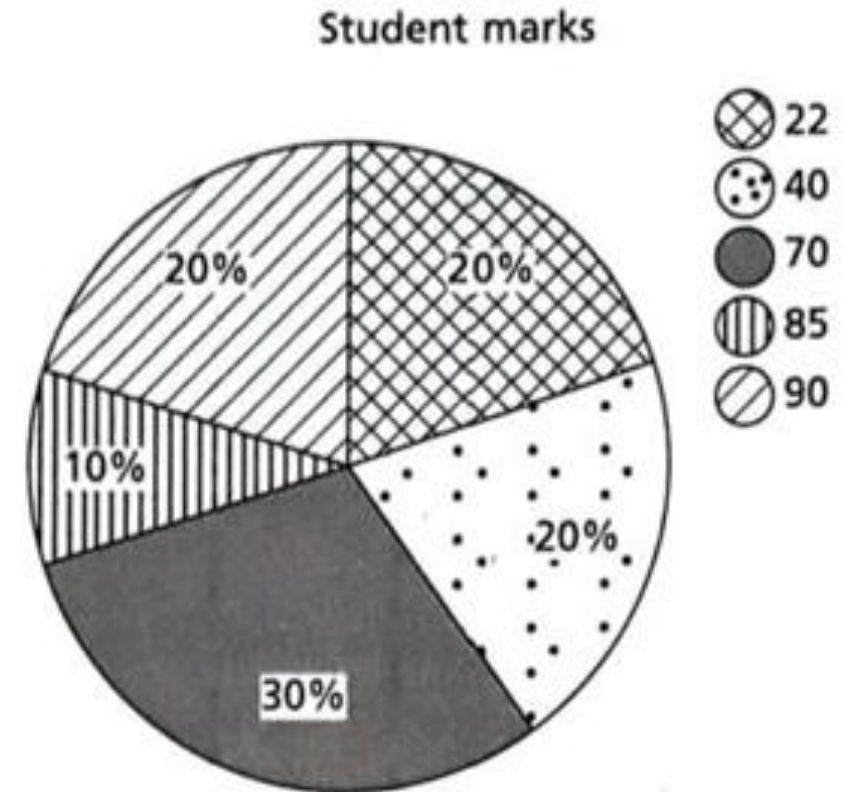


# Pie Chart

- Pie charts are also effective in representing univariate data. They illustrate the percentage distribution of different categories within a dataset.
- Figure presents the percentage frequency distribution of students' marks {22, 22, 40, 40, 70, 70, 70, 85, 90}.
- It can be observed that **2 students** scored **22 marks** out of a total of **10 students**. The proportion is calculated as:

$$\frac{2}{10} \times 100 = 20\%$$

- Thus, **20%** of the pie chart is allocated to students scoring **22 marks**, as shown in Figure.



## Pie Chart Example ▾



Ru

```
import matplotlib.pyplot as plt
```

```
# Sample dataset
```

```
companies = ['Company A', 'Company B', 'Company C', 'Company D', 'Company E']
```

```
market_share = [30, 25, 20, 15, 10]
```

```
# Creating the pie chart
```

```
plt.pie(market_share, labels=companies, autopct='%1.1f%%', colors=['blue', 'green', 'red', 'purple', 'orange'], startangle=140)
```

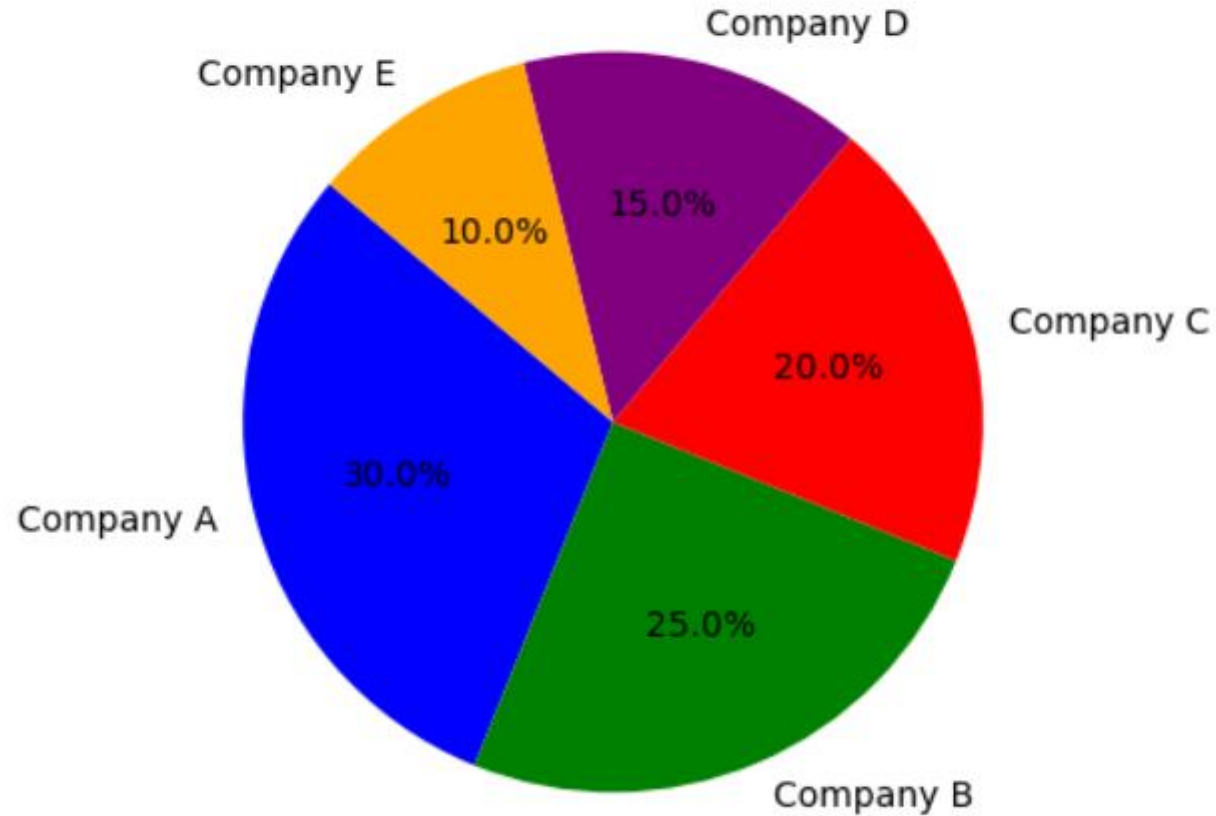
```
# Adding title
```

```
plt.title('Market Share Distribution')
```

```
# Displaying the chart
```

```
plt.show()
```

## Market Share Distribution



# Benefits of a Pie Chart

A **Pie Chart** is a circular statistical graphic that represents data as proportional slices. Here are its key benefits:

Benefit	Description
<b>Easy to Understand</b>	A pie chart visually represents proportions, making it simple to interpret at a glance.
<b>Effective for Comparisons</b>	It helps compare relative sizes of different categories within a dataset.
<b>Shows Part-to-Whole Relationship</b>	Clearly illustrates how each category contributes to the total.
<b>Visually Appealing</b>	Enhances data presentation in reports, presentations, and dashboards.
<b>Highlights Dominant Categories</b>	Larger slices stand out, making it easy to identify significant segments.
<b>Customization Options</b>	Can include labels, colors, and percentages for better clarity.
<b>Widely Used in Business &amp; Analytics</b>	Commonly used for market share, budget distribution, and sales analysis.

# When to Use a Pie Chart:

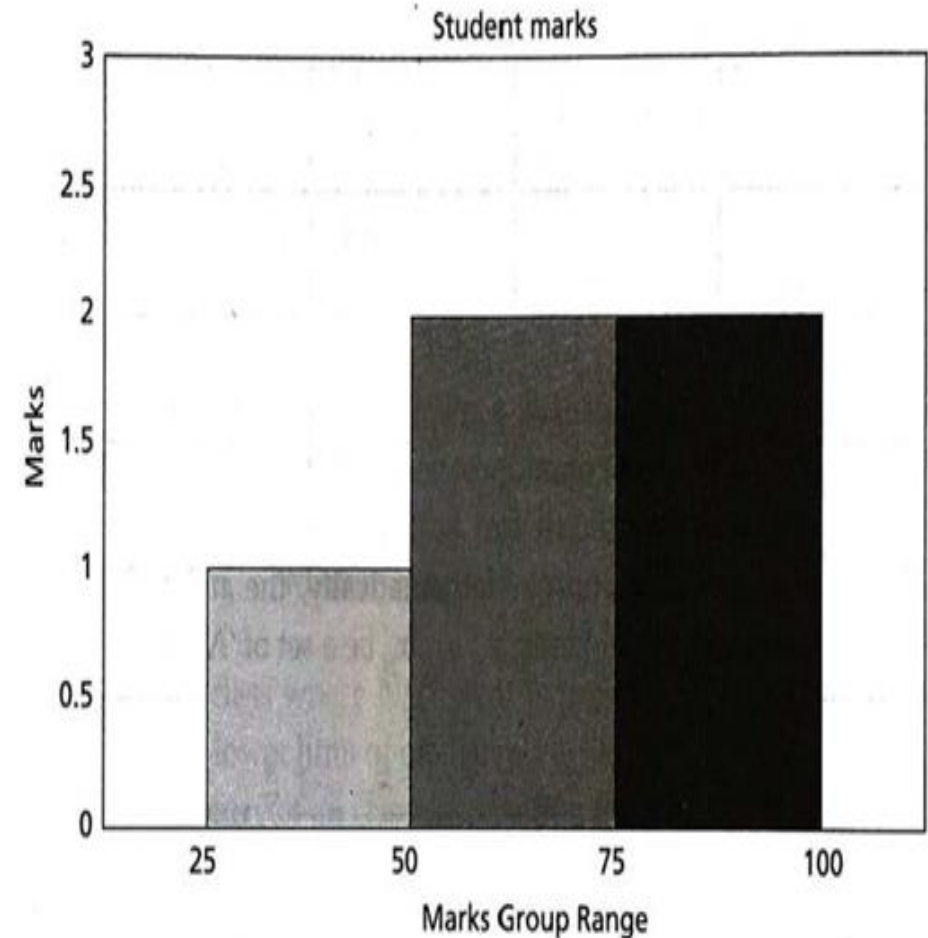
- When you need to show proportions of different categories.
- When there are limited (3-7) categories for clear visualization.
- When comparing percentages or shares within a whole.

# Histogram

- Histograms play a crucial role in data mining by visualizing frequency distributions.
- Unlike bar charts, histograms group **continuous data into intervals**.

# Histogram

- The histogram for students' marks {45, 60, 60, 80, 85}, categorized into the following ranges:
  - 0–25
  - 26–50
  - 51–75
  - 76–100
- is depicted in Figure .
- From the histogram, it is evident that **2 students fall within the 76–100 range.**



# Histogram

- Histograms provide key insights, including:
  - The **nature** of the dataset
  - The **mode**, which represents the peak frequency
  - The **shape**, indicating skewness and distribution patterns
- Histograms are useful for understanding data frequency, trends, and patterns.

## Skewness Indicators:

- **No skewness:** Mean  $\approx$  Median  $\approx$  Mode
- **Right skew:** Mean  $>$  Median  $>$  Mode
- **Left skew:** Mean  $<$  Median  $<$  Mode

# Histogram

```
import matplotlib.pyplot as plt
import numpy as np

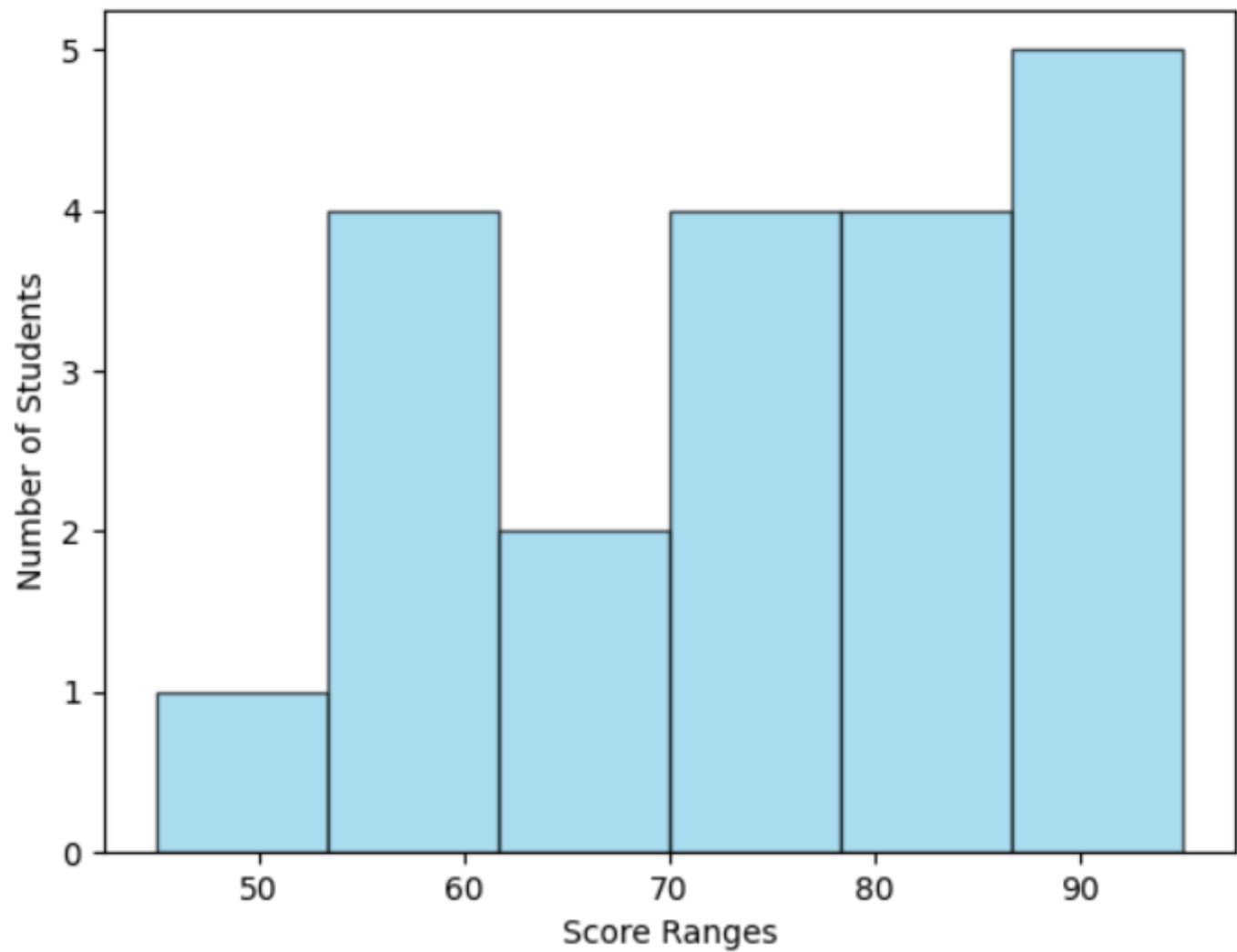
# Sample dataset representing exam scores of students
scores = [55, 89, 76, 65, 90, 45, 78, 88, 92, 56, 73, 81, 67, 80, 85, 60, 77, 83, 95, 59]

# Creating the histogram
plt.hist(scores, bins=6, color='skyblue', edgecolor='black', alpha=0.7)

# Adding titles and labels
plt.xlabel('Score Ranges')
plt.ylabel('Number of Students')
plt.title('Distribution of Exam Scores')

# Displaying the chart
plt.show()
```

Distribution of Exam Scores



```
import matplotlib.pyplot as plt
import numpy as np

# Sample dataset
categories = ['A', 'B', 'C', 'D', 'E']
values = [10, 20, 15, 25, 18]

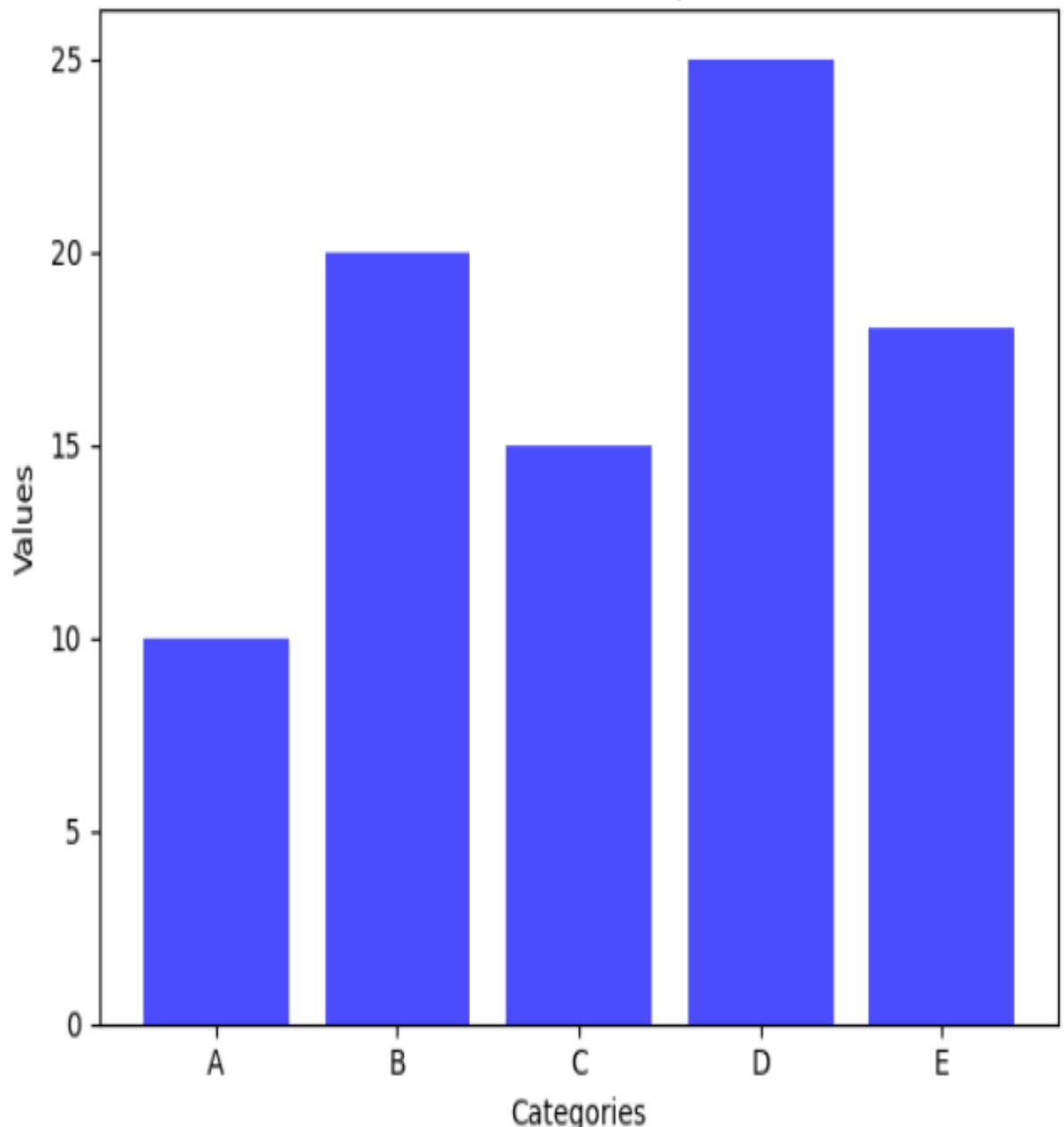
# Creating a bar chart
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.bar(categories, values, color='blue', alpha=0.7)
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Chart Example')

# Sample dataset for histogram
data = np.random.normal(50, 15, 1000) # Generating random data

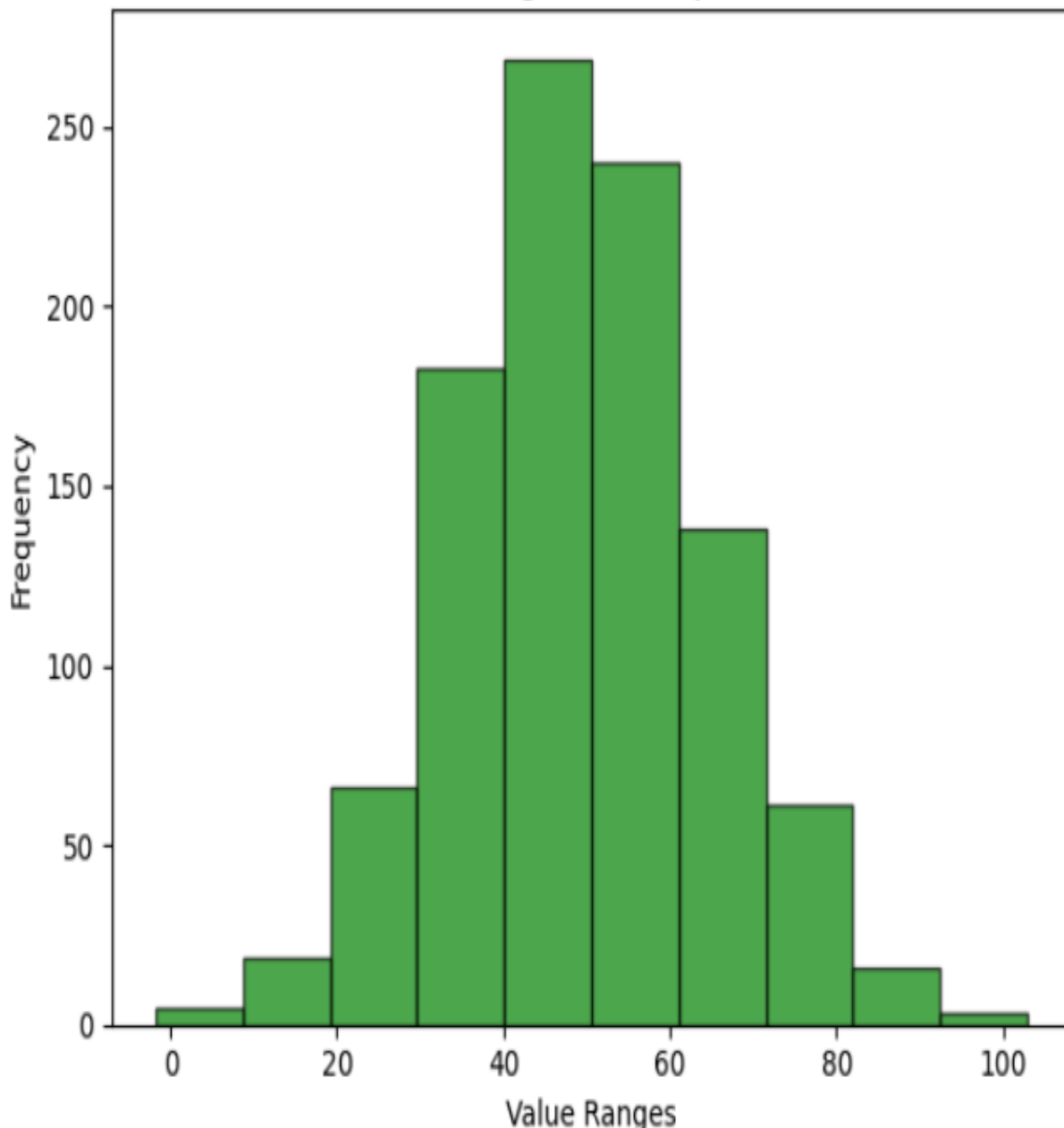
# Creating a histogram
plt.subplot(1, 2, 2)
plt.hist(data, bins=10, color='green', edgecolor='black', alpha=0.7)
plt.xlabel('Value Ranges')
plt.ylabel('Frequency')
plt.title('Histogram Example')

# Displaying the charts
plt.tight_layout()
plt.show()
```

Bar Chart Example



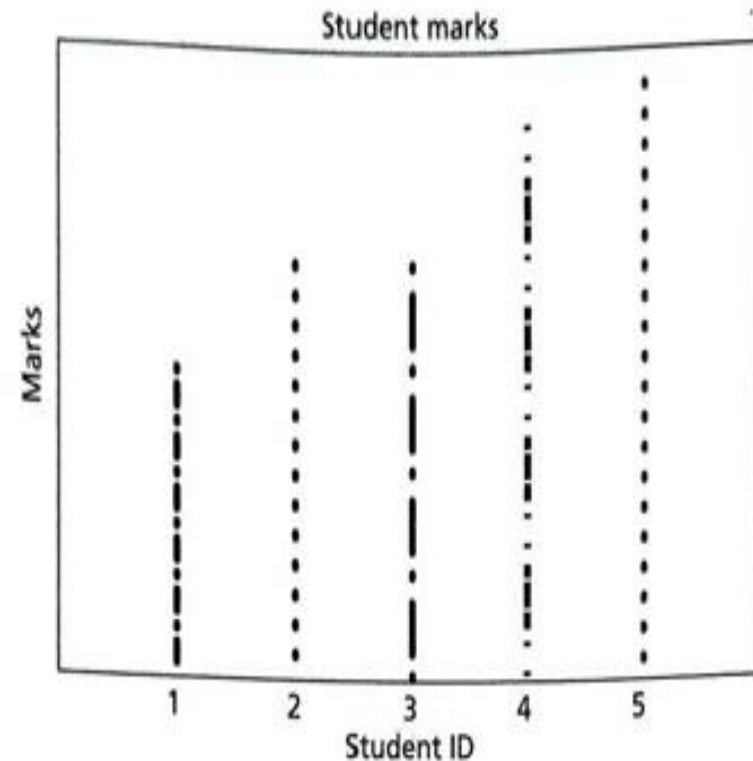
Histogram Example



Feature	Bar Chart	Histogram
<b>Definition</b>	A bar chart represents categorical data using bars of different heights.	A histogram represents the distribution of numerical data by grouping it into bins.
<b>Data Type</b>	Categorical (e.g., categories, groups, labels).	Continuous (e.g., age, height, exam scores).
<b>Bars Separation</b>	Bars are separated by gaps to distinguish categories.	Bars are adjacent without gaps, indicating continuous data.
<b>X-Axis Representation</b>	Represents distinct, separate categories.	Represents continuous data intervals (bins).
<b>Y-Axis Representation</b>	Represents frequency, counts, or another metric for each category.	Represents the number of data points falling within each interval.
<b>Usage</b>	Used for comparing different categories.	Used for analyzing data distribution and frequency.
<b>Example</b>	Comparing sales of different products.	Analyzing the age distribution of a group of people.

# Dot Plots

- Dot plots are similar to bar charts but display individual data points instead of bars. They provide a clearer representation with less clutter.
- The dot plot for English marks of five students (**Student IDs: 1, 2, 3, 4, 5**) with scores **{45, 60, 60, 80, 85}** is shown in Figure .
- A key advantage of dot plots is that they allow easy visual inspection to determine which student scored higher.



# Central Tendency

- Since it is impractical to remember all data points, summarizing the dataset is essential for effective analysis. One such summary measure is **central tendency**, which helps describe data characteristics and enables comparisons.
- Central tendency represents values where data points tend to cluster, typically near the **center** of a dataset. The three most common measures of central tendency are:
  - **Mean**
  - **Median**
  - **Mode**

# Mean (Arithmetic Average)

- The **mean** is the most commonly used measure of central tendency. It represents the **center** of a dataset and is often referred to as the **average**. The mean is calculated by summing all values and dividing by the number of observations.
- Mathematically, for a dataset with **N** values ( $x_1, x_2, \dots, x_n$ ), the mean is given by:

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

For example, the mean of three numbers 10, 20, and 30 is:

$$\bar{x} = \frac{10 + 20 + 30}{3} = \frac{60}{3} = 20$$

# Weighted Mean

- Unlike the arithmetic mean, which assigns equal importance to all values, the **weighted mean** assigns different weights based on the importance of each item. This allows more significance to be given to specific values when necessary.
- In the case of a **frequency distribution**, the mid-values of each range are taken for computation.
- For **weighted mean**, the mean is calculated by multiplying the proportion by the corresponding group mean and summing the results. This method is mainly used when sample sizes are unequal.

# Weighted Mean

- The **weighted mean** (or weighted average) is a type of average where each value in the data set is multiplied by a corresponding weight, reflecting its relative importance or frequency, before calculating the average.

The formula for the weighted mean is:

$$\text{Weighted Mean} = \frac{\sum(x_i \cdot w_i)}{\sum w_i}$$

Where:

- $x_i$  is each individual data value.
- $w_i$  is the weight corresponding to each data value.
- $\sum(x_i \cdot w_i)$  is the sum of the products of each value and its weight.
- $\sum w_i$  is the sum of all the weights.

# Weighted Mean

## Example:

Suppose you have the following data set of test scores and their corresponding weights (e.g., the number of hours spent studying for each test):

Test Score ( $x_i$ )	Weight ( $w_i$ )
90	3
80	2
70	1

To find the weighted mean:

1. Multiply each score by its weight:

$$90 \times 3 = 270$$

$$80 \times 2 = 160$$

$$70 \times 1 = 70$$

# Weighted Mean

2. Sum the products:

$$270 + 160 + 70 = 500$$

3. Sum the weights:

$$3 + 2 + 1 = 6$$

4. Calculate the weighted mean:

$$\text{Weighted Mean} = \frac{500}{6} = 83.33$$

So, the weighted mean score is approximately **83.33**.

# Geometric Mean

Let  $x_1, x_2, \dots, x_n$  be a set of  $N$  values or observations. The **geometric mean** is the  $N^{\text{th}}$  root of the product of all items. It is given by the formula:

$$\text{Geometric Mean} = \left( \prod_{i=1}^N x_i \right)^{\frac{1}{N}} = \sqrt[N]{x_1 \times x_2 \times \dots \times x_N}$$

Where:

- $n$  = number of items
- $x_i$  = values of items

For example, if the values are **6 and 8**, the geometric mean is:

$$\sqrt{6 \times 8} = \sqrt{48}$$

# Geometric Mean

For large datasets, computing the geometric mean directly is difficult. Instead, it is usually calculated using logarithms:

$$\text{Anti-log} \left( \frac{\log x_1 + \log x_2 + \dots + \log x_N}{N} \right)$$

Or,

$$\text{Anti-log} \left( \frac{\sum_{i=1}^N \log x_i}{N} \right)$$

# Logarithm (Log)

A **logarithm** is the inverse operation to exponentiation. In simple terms, it answers the question: "To what power must we raise a certain base to obtain a given number?"

The general form of a logarithmic equation is:

$$\log_b(x) = y$$

Where:

- $b$  is the **base** of the logarithm.
- $x$  is the **argument** or **input**.
- $y$  is the **exponent** or **result**, such that  $b^y = x$ .

For example:

- $\log_2(8) = 3$ , because  $2^3 = 8$ .
- $\log_{10}(1000) = 3$ , because  $10^3 = 1000$ .

The most common types of logarithms are:

- **Common logarithms** (base 10):  $\log_{10}(x)$ , often written as  $\log(x)$ .
- **Natural logarithms** (base  $e$ ):  $\ln(x)$ , where  $e$  is approximately 2.718.

## Antilogarithm (Anti-Log)

An **antilogarithm** is the inverse of taking the logarithm. It answers the question: **"What number do I get by raising the base of the logarithm to a certain power?"**

In other words, if  $y = \log_b(x)$ , then the antilogarithm is:

$$x = b^y$$

For example:

- If  $\log_{10}(x) = 3$ , then  $x = 10^3 = 1000$ .
- If  $\log_2(x) = 5$ , then  $x = 2^5 = 32$ .

# Limitation of Mean

- One limitation of the **mean** is its sensitivity to extreme values.
- Even small changes in input can drastically affect the result.
- To mitigate this, the **top 2% of values are often removed** before calculating the mean for larger datasets.

## 2. Median

- The **median** is the middle value in an ordered dataset.
- If the total number of observations is **odd**, the median is the middle value.
- If the total number of observations is **even**, the median is the average of the two central values.
- The median effectively **divides the dataset into two equal halves**, where half the values are lower and half are higher than the median.

## 2. Median

In grouped data, the **median class** is the class where the  $N/2^{\text{th}}$  item appears. The formula for calculating the median in **continuous data** is:

$$\text{Median} = L + \left( \frac{N}{2} - cf \right) \times \frac{i}{f}$$

Where:

- $L$  = lower limit of the median class
- $N$  = total number of observations
- $cf$  = cumulative frequency of classes preceding the median class
- $f$  = frequency of the median class
- $i$  = class width

# 3. Mode

The **mode** is the value that appears most frequently in a dataset. It represents the **most common observation**.

- The mode is primarily used for **discrete data**.
- For **continuous data**, mode is not applicable because no values repeat exactly.

The mode is determined by calculating the **frequency of each value**, with the value having the **highest frequency** being the mode.

# Datasets are classified based on the number of modes:

- Unimodal (one mode)
- Bimodal (two modes)
- Trimodal (three modes)

## 1. Unimodal (One Mode) Data

- **Definition:** A unimodal data set has only **one mode**. In other words, there is a single value that appears more frequently than any other value.
- **Example:**
  - Data: 4, 5, 6, 7, 7, 8, 9
  - The mode is 7 because it appears twice, while all other values appear only once.

## 2. Bimodal (Two Modes) Data

- **Definition:** A bimodal data set has **two modes**. This means there are two values that appear with the highest frequency and both occur more frequently than other values in the data set.
- **Example:**
  - Data: 1, 2, 3, 3, 4, 4, 5
  - The modes are 3 and 4, as they both appear twice, which is more frequent than the other values.

### 3. Trimodal (Three Modes) Data

- **Definition:** A trimodal data set has **three modes**. This means there are three values that appear with the highest frequency and they each occur more frequently than the other values.
- **Example:**
  - Data: 1, 2, 2, 3, 3, 4, 4, 4, 5
  - The modes are 2, 3, and 4, because each of these values appears three times, more frequently than other values.

# Dispersion

- **Dispersion** refers to the spread of a dataset around its central tendency (mean, median, or mode).
- It is measured using various statistical methods, including **range, variance, standard deviation, and interquartile range**.
- These are second-order measures of data variability.

# Range

- The **range** is the difference between the maximum and minimum values in a dataset:
- $\text{Range} = \max(x) - \min(x)$

# Standard Deviation

- The **mean** alone does not describe the entire dataset accurately.
- For example, the datasets **(10, 20, 30)** and **(10, 50, 0)** both have a mean of **20**, but their spreads differ significantly.
- The **standard deviation ( $\sigma$ )** measures the average deviation of each data point from the mean.
- It is calculated using the formula:

# Standard Deviation

$$\sigma = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N - 1}}$$

Where:

- $N$  = total number of observations
- $x_i$  = individual data values
- $\bar{x}$  = mean of the dataset

In real-world applications,  $N - 1$  is used instead of  $N$  in the denominator because it provides a more accurate estimate.

```
import numpy as np
import statistics as stats

# Sample dataset
data = [10, 20, 20, 30, 30, 30, 40, 50, 60]
weights = [1, 2, 2, 3, 3, 3, 4, 5, 6] # Weights for weighted mean

# Mean
mean = np.mean(data)
print(f"Mean: {mean}")

# Weighted Mean
weighted_mean = np.average(data, weights=weights)
print(f"Weighted Mean: {weighted_mean}")
```

```
# Median
```

```
median = np.median(data)  
print(f"Median: {median}")
```

```
# Mode
```

```
mode = stats.mode(data)  
print(f"Mode: {mode}")
```

```
# Standard Deviation
```

```
std_dev = np.std(data, ddof=1) # Sample standard deviation  
print(f"Standard Deviation: {std_dev}")
```

Mean: 32.22222222222222

---

Weighted Mean: 38.96551724137931

---

Median: 30.0

---

Mode: 30

---

Standard Deviation: 15.634719199411432

# Quartiles and Interquartile Range (IQR)

- Quartiles divide a dataset into **four equal parts**:
  - $Q_1$  (First Quartile) = 25th percentile
  - $Q_2$  (Second Quartile or Median) = 50th percentile
  - $Q_3$  (Third Quartile) = 75th percentile
- The **Interquartile Range (IQR)** measures dispersion by finding the difference between the **third quartile and the first quartile**:
- **$IQR = Q_3 - Q_1$**
- This measure is useful for detecting **outliers**, which are values that deviate significantly from the rest of the data.
- A common rule is that **outliers** are values at least  **$1.5 \times IQR$**  above  **$Q_3$**  or below  **$Q_1$** .

# Example : Finding IQR

- **Dataset:** Patients' ages = {12, 14, 19, 22, 24, 26, 28, 31, 34}

## **Solution:**

1. Find the median ( $Q_2$ ):

- The dataset has 9 values, so the 5th value is the median.
- Median ( $Q_2$ ) = 24

2. Find the first quartile ( $Q_1$ ):

- The lower half of the dataset: {12, 14, 19, 22}
- The median of this subset (average of 2nd and 3rd values):

$$Q_1 = \frac{14 + 19}{2} = 16.5$$

# Example : Finding IQR

3. Find the third quartile ( $Q_3$ ):

- The upper half of the dataset: {26, 28, 31, 34}
- The median of this subset (average of 7th and 8th values):

$$Q_3 = \frac{28 + 31}{2} = 29.5$$

4. Calculate IQR:

$$IQR = Q_3 - Q_1 = 29.5 - 16.5 = 13$$

Thus, the IQR = 13.

# Semi Interquartile Range (SIQR)

The semi-interquartile range (SIQR) is half of the interquartile range (IQR) and is calculated as:

$$\begin{aligned} \text{SIQR} &= \frac{1}{2} \times \text{IQR} \\ &= \frac{1}{2} \times 13 = 6.5 \end{aligned}$$

```
import numpy as np

# Sample dataset
data = [12, 15, 14, 10, 18, 20, 22, 25, 30, 35]

# Calculate quartiles
Q1 = np.percentile(data, 25)
Q2 = np.percentile(data, 50) # Median
Q3 = np.percentile(data, 75)

# Compute Interquartile Range (IQR)
IQR = Q3 - Q1
```

```
# Detect Outliers
```

```
lower_bound = Q1 - 1.5 * IQR
```

```
upper_bound = Q3 + 1.5 * IQR
```

```
outliers = [x for x in data if x < lower_bound or x > upper_bound]
```

```
# Display results
```

```
print(f"Q1 (First Quartile) : {Q1}")
```

```
print(f"Q2 (Median) : {Q2}")
```

```
print(f"Q3 (Third Quartile) : {Q3}")
```

```
print(f"Interquartile Range : {IQR}")
```

```
print(f"Outliers : {outliers}")
```

```
import numpy as np

# Sample dataset
data = [12, 15, 14, 10, 18, 20, 22, 25, 30, 35]

# Calculate quartiles
Q1 = np.percentile(data, 25)
Q2 = np.percentile(data, 50) # Median
Q3 = np.percentile(data, 75)

# Compute Interquartile Range (IQR)
IQR = Q3 - Q1

# Detect Outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
outliers = [x for x in data if x < lower_bound or x > upper_bound]

# Display results
print(f"Q1 (First Quartile) : {Q1}")
print(f"Q2 (Median)          : {Q2}")
print(f"Q3 (Third Quartile)   : {Q3}")
print(f"Interquartile Range    : {IQR}")
print(f"Outliers                 : {outliers}")
```

Q1 (First Quartile) : 14.25  
Q2 (Median) : 19.0  
Q3 (Third Quartile) : 24.25  
Interquartile Range : 10.0  
Outliers : []

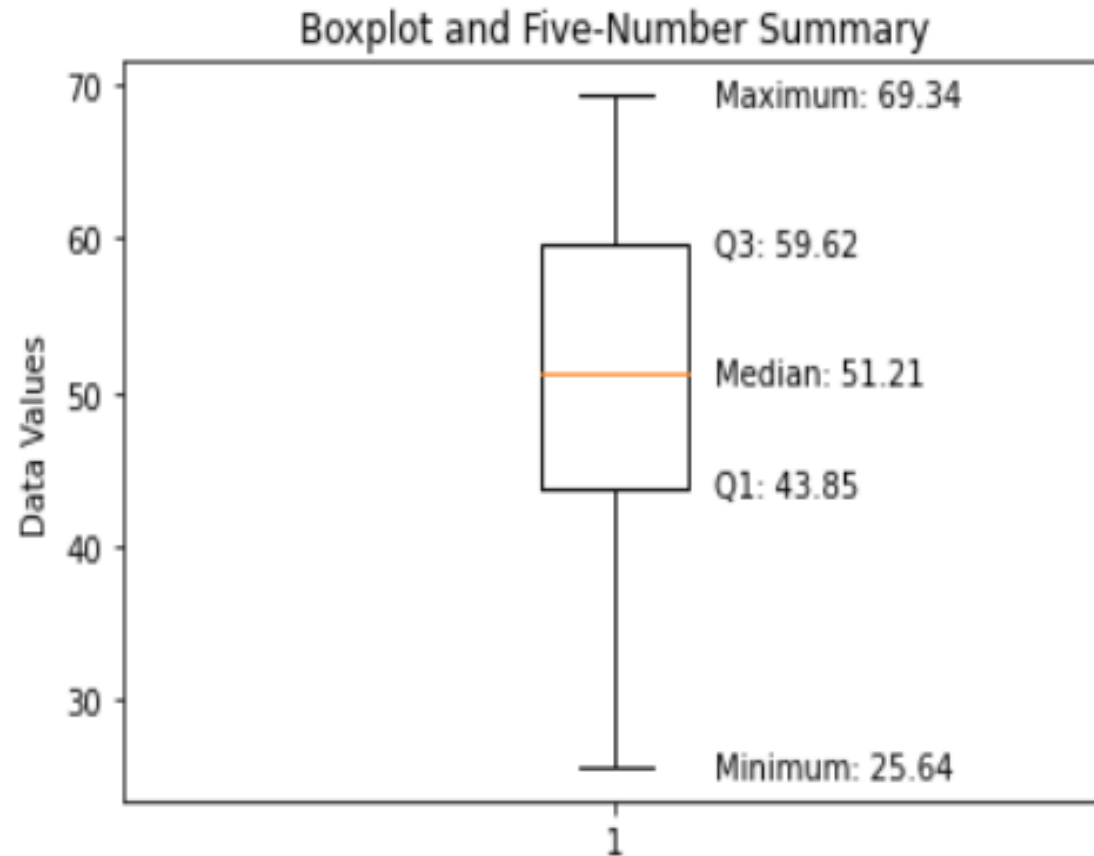
# Five-Point Summary and Box Plots

A dataset's **five-point summary** consists of the **minimum**, **first quartile ( $Q_1$ )**, **median ( $Q_2$ )**, **third quartile ( $Q_3$ )**, and **maximum** values, written in the order:

Minimum,  $Q_1$ , Median,  $Q_3$ , Maximum

**Box plots** (also known as **box-and-whisker plots**) visually represent this summary and are useful for displaying **data distributions**. Box plots can be used for **continuous variables** and **nominal variables** to summarize data effectively.

# Five-Point Summary and Box Plots



# Key Features of a Box Plot:

- The box contains the bulk of the data (**between  $Q_1$  and  $Q_3$** ).
- The line inside the box represents the median ( $Q_2$ ).
- If the median is not equidistant between  **$Q_1$  and  $Q_3$** , **the dataset is skewed**.
- **Whiskers extend** from both ends of the box, indicating the spread of the tails and the range (minimum and maximum values).

# Example 2.5: Finding the Five-Point Summary

- **Dataset:** {13, 11, 2, 3, 4, 8, 9}

## **Solution:**

1. Find the minimum and maximum values:

- Minimum = 2, Maximum = 13

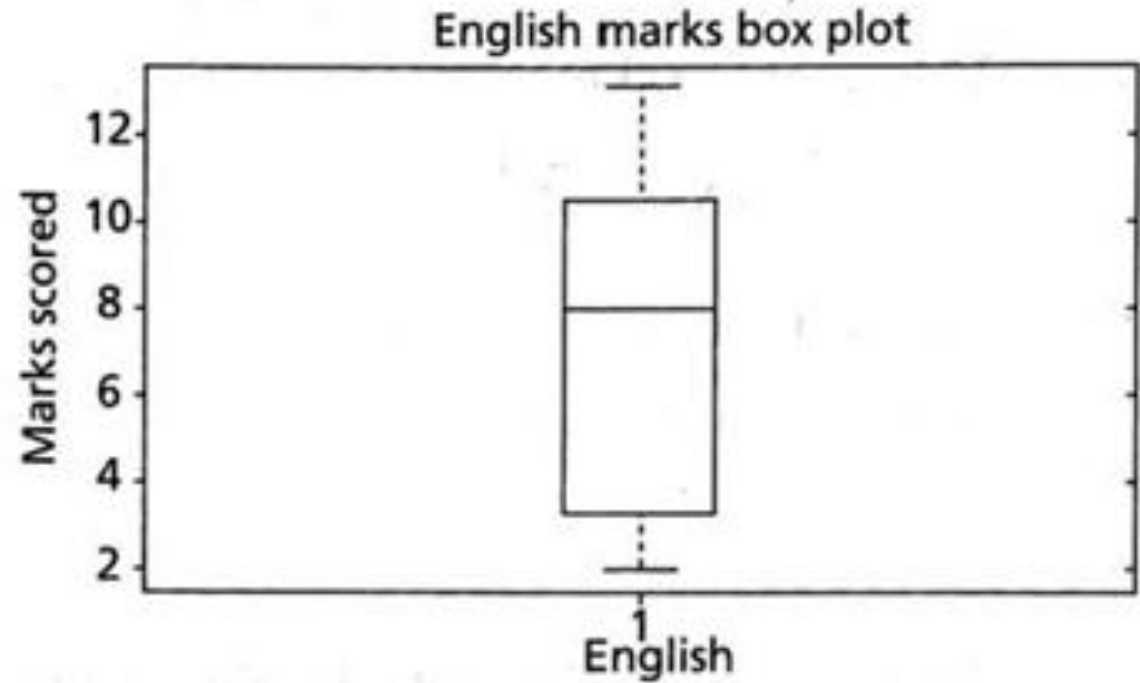
2. Find the quartiles ( $Q_1$ ,  $Q_2$ ,  $Q_3$ ):

- $Q_1 = 3$ , Median ( $Q_2$ ) = 8,  $Q_3 = 11$

3. Five-point summary:

(2, 3, 8, 11, 13) or (Minimum,  $Q_1$ , Median,  $Q_3$ , Maximum)

- A **box plot** is an effective way to visualize this summary. The corresponding box plot for this dataset is shown in **Figure 2.7**.



# Shape

- The **shape** of a dataset describes its **symmetry** or **asymmetry** and the **peak location** of its distribution.
- **Skewness: Skewness** measures the **direction and degree of symmetry** in a dataset. It is a **third-order moment measure** that determines whether the dataset is **symmetrical** or **skewed**.
  - A perfectly normal distribution has **zero skewness**.
  - Real-world datasets may exhibit **positive** or **negative skewness** depending on their distribution.

# Skewness

A dataset may exhibit **positive skewness** or **negative skewness** based on the distribution of values:

- **Positive Skewness:** When a dataset contains **higher values** concentrated towards the right, with a longer **right tail**, it is **positively skewed**.
- **Negative Skewness:** When a dataset contains **lower values** concentrated towards the left, with a longer **left tail**, it is **negatively skewed**.

# Skewness

- The **distribution of data** affects the relationship between the **mean** and **median**:
  - In a **positively skewed dataset**, the **mean is greater than the median**.
  - In a **negatively skewed dataset**, the **median is greater than the mean**.
- This skewness can impact **data mining algorithms**, as skewed data may contain more **outliers** that affect the model's performance.

# Pearson's Skewness Coefficient

The relationship between skewness, mean, and median can be estimated using **Pearson's Skewness Coefficient**:

$$\text{Skewness} = \frac{3(\mu - \text{median})}{\sigma}$$

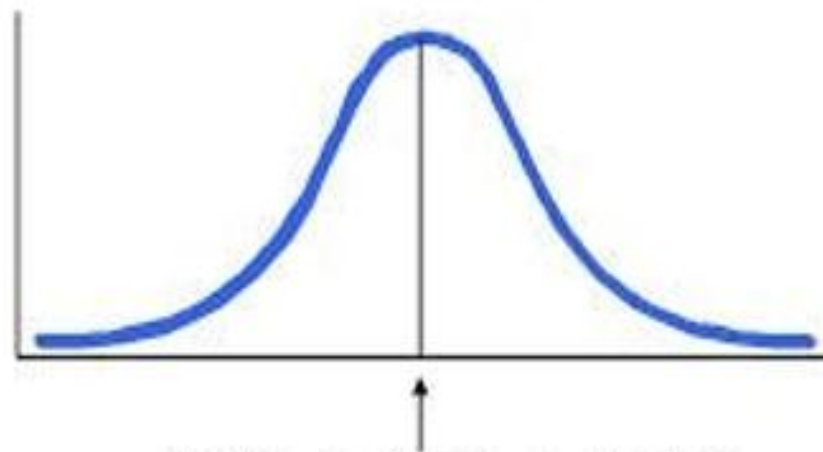
Alternatively, skewness can be calculated using the following formula:

$$\frac{1}{N} \sum_{i=1}^N \left( \frac{x_i - \mu}{\sigma} \right)^3$$

where:

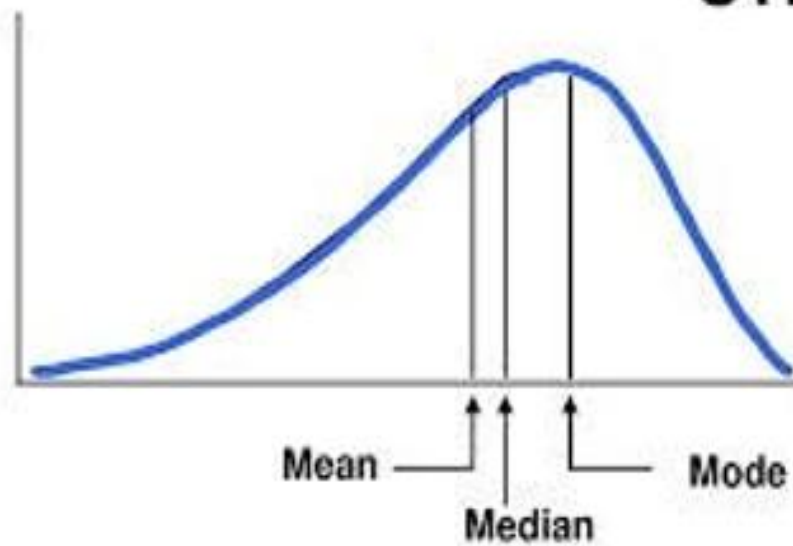
- $\mu$  is the **population mean**
- $\sigma$  is the **standard deviation** of the dataset
- **N** is the total number of observations

For **bias correction**, **N - 1** is sometimes used instead of **N**.

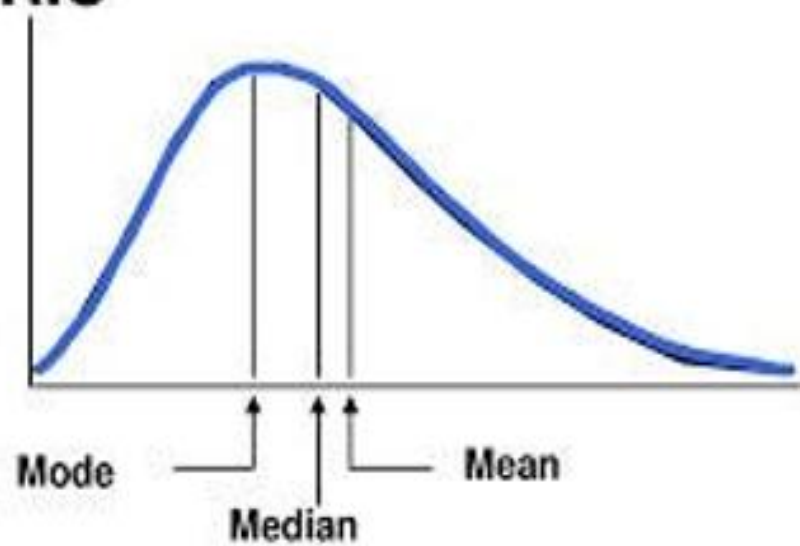


Mode = Mean = Median

**SYMMETRIC**



**SKEWED LEFT**  
(negatively)



**SKEWED RIGHT**  
(positively)

# Kurtosis

- **Kurtosis** measures the **peakedness** of a data distribution:
  - A **high kurtosis** value indicates a dataset with **sharp peaks and heavy tails**.
  - A **low kurtosis** value suggests a dataset with a **flat distribution and shorter tails**.
- In a **normal distribution**, the dataset has a **bell-shaped curve** with **moderate tails**. Low kurtosis means that the dataset has **fewer or no outliers**.

# Formula for Kurtosis

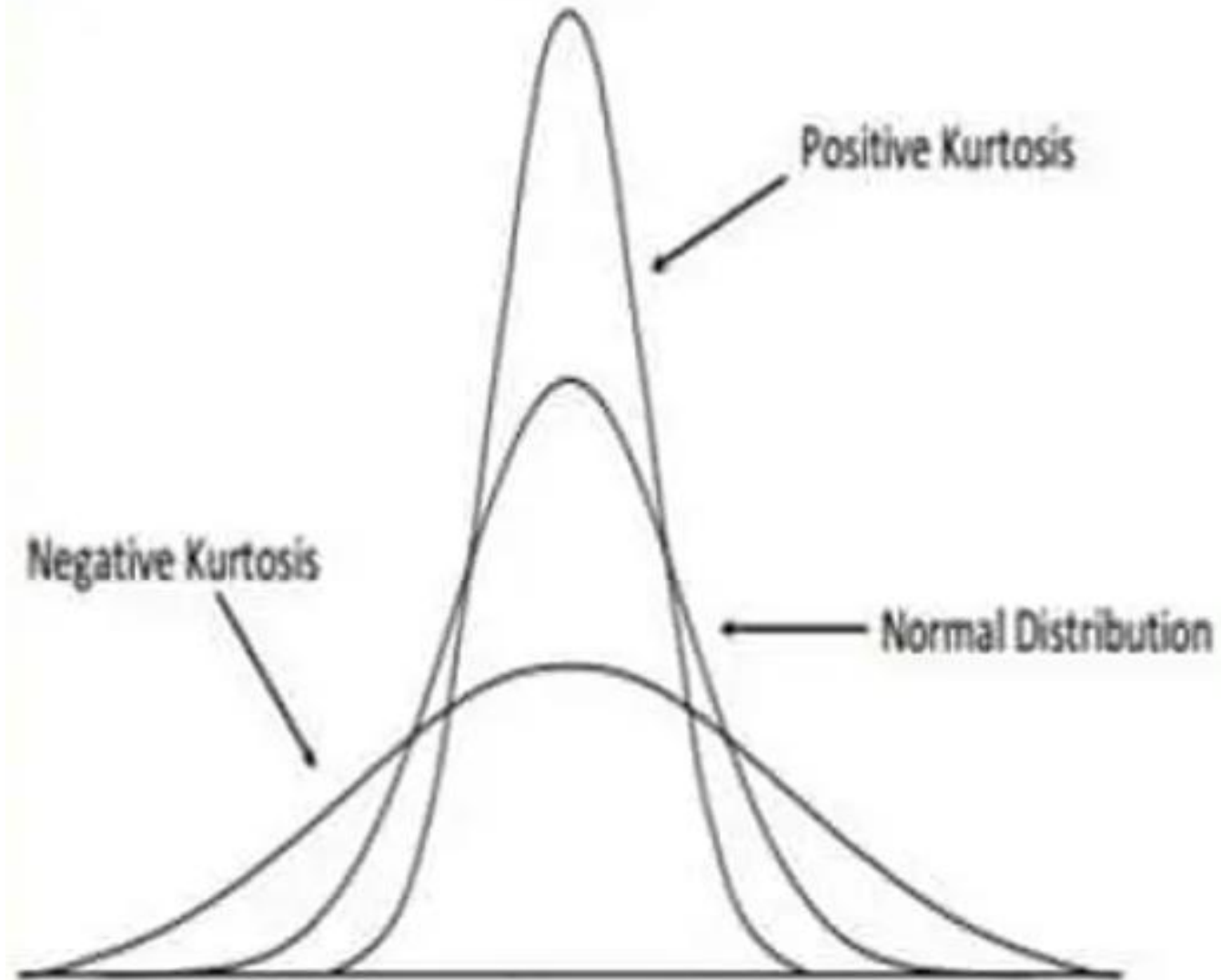
$$\frac{1}{N} \sum_{i=1}^N \left( \frac{x_i - \bar{x}}{\sigma} \right)^4$$

where:

- $\bar{x}$  is the **mean**
- $\sigma$  is the **standard deviation**
- **N** is the **number of values in the dataset**

For correction, **N - 1** is sometimes used in the numerator instead of **N**.

# Skewness and Kurtosis



# Mean Absolute Deviation (MAD)

- Mean Absolute Deviation (MAD) is a **robust dispersion measure** that is **resistant to outliers**. It is computed by finding the absolute deviation of each data point from the **mean** and then averaging these deviations.

The sum of absolute deviations is given by:

$$\sum |x - \mu|$$

where:

- $x$  represents each data point
- $\mu$  is the **mean** of the dataset
- $N$  is the total number of observations

# Coefficient of Variation (CV)

The **coefficient of variation (CV)** is used for comparing datasets with **different units** or **different scales**. It is defined as the **ratio of standard deviation to mean**, expressed as a percentage:

$$CV = \frac{\sigma}{\mu} \times 100\%$$

CV is particularly useful for assessing the **relative variability** between datasets.

# Special Univariate Plots

A **Stem-and-Leaf Plot** is a useful method to visualize the **shape** and **distribution** of a dataset. Each data point is split into two parts:

- **Stem** → Represents the leading digits
- **Leaf** → Represents the last digit

For example, the number **45** is divided into:

- **Stem = 4**
- **Leaf = 5**

# Example: English Subject Marks

- Consider the dataset of English subject marks: **45, 60, 60, 80, 85**. The **stem-and-leaf plot** for this dataset is:

Stem	Leaf
4	5
5	
6	0 0
7	
8	0 5

# From the plot, we observe that:

- The first column represents the stem.
- The second column represents the leaf.
- For 60 marks, two students scored 60, represented by two leaves (0,0) under stem 6.
- For 80 marks, one student scored 80 and another scored 85.

# Normal Distribution

- The **Normal Distribution** is one of the most important probability distributions in statistics and data analysis.
- It is commonly referred to as the **Gaussian distribution** after the mathematician Carl Friedrich Gauss.
- This distribution is widely used because it describes many natural phenomena, such as heights, weights, test scores, and measurement errors.

# Characteristics of Normal Distribution:

- 1. Symmetry:** The normal distribution is symmetric around its mean, meaning the left and right sides of the distribution are mirror images of each other.
- 2. Bell-shaped curve:** The probability density function (PDF) of a normal distribution forms a bell-shaped curve.
- 3. Mean, Median, and Mode are Equal:** In a perfectly normal distribution, the mean, median, and mode are all the same and located at the center of the distribution.

4. Spread determined by the Standard Deviation: The spread of the normal distribution is controlled by its standard deviation ( $\sigma$ ). A larger standard deviation means the distribution is wider, while a smaller standard deviation means the distribution is more concentrated around the mean.
5. 68-95-99.7 Rule (Empirical Rule):
  - About 68% of the data falls within 1 standard deviation from the mean.
  - About 95% of the data falls within 2 standard deviations from the mean.
  - About 99.7% of the data falls within 3 standard deviations from the mean.

# Mathematical Formula of Normal Distribution:

- The probability density function (PDF) for a normal distribution is given by the formula:

$$f(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Where:

- $x$  is a value from the dataset.
- $\mu$  is the **mean** of the distribution.
- $\sigma$  is the **standard deviation**.
- $\pi$  is a constant ( $\sim 3.14159$ ).
- $e$  is Euler's number ( $\sim 2.71828$ ).

# Example: Normal Distribution of Heights

- Suppose we have data on the heights of adult women in a population, and we know that:
  - The mean height ( $\mu$ ) is 65 inches.
  - The standard deviation ( $\sigma$ ) is 3 inches.

We can model this data using a normal distribution.

**1. Plotting the Normal Distribution :** The normal distribution with mean 65 and standard deviation 3 would have a bell-shaped curve centered around 65 inches.

## **2. Interpreting the 68-95-99.7 Rule**

For this distribution:

- 1. 68%** of the adult women in this population have heights between **62 and 68 inches** (i.e., within one standard deviation of the mean).
- 2. 95%** of them have heights between **59 and 71 inches** (i.e., within two standard deviations).
- 3. 99.7%** of them have heights between **56 and 74 inches** (i.e., within three standard deviations).

# Python Code Example:

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats

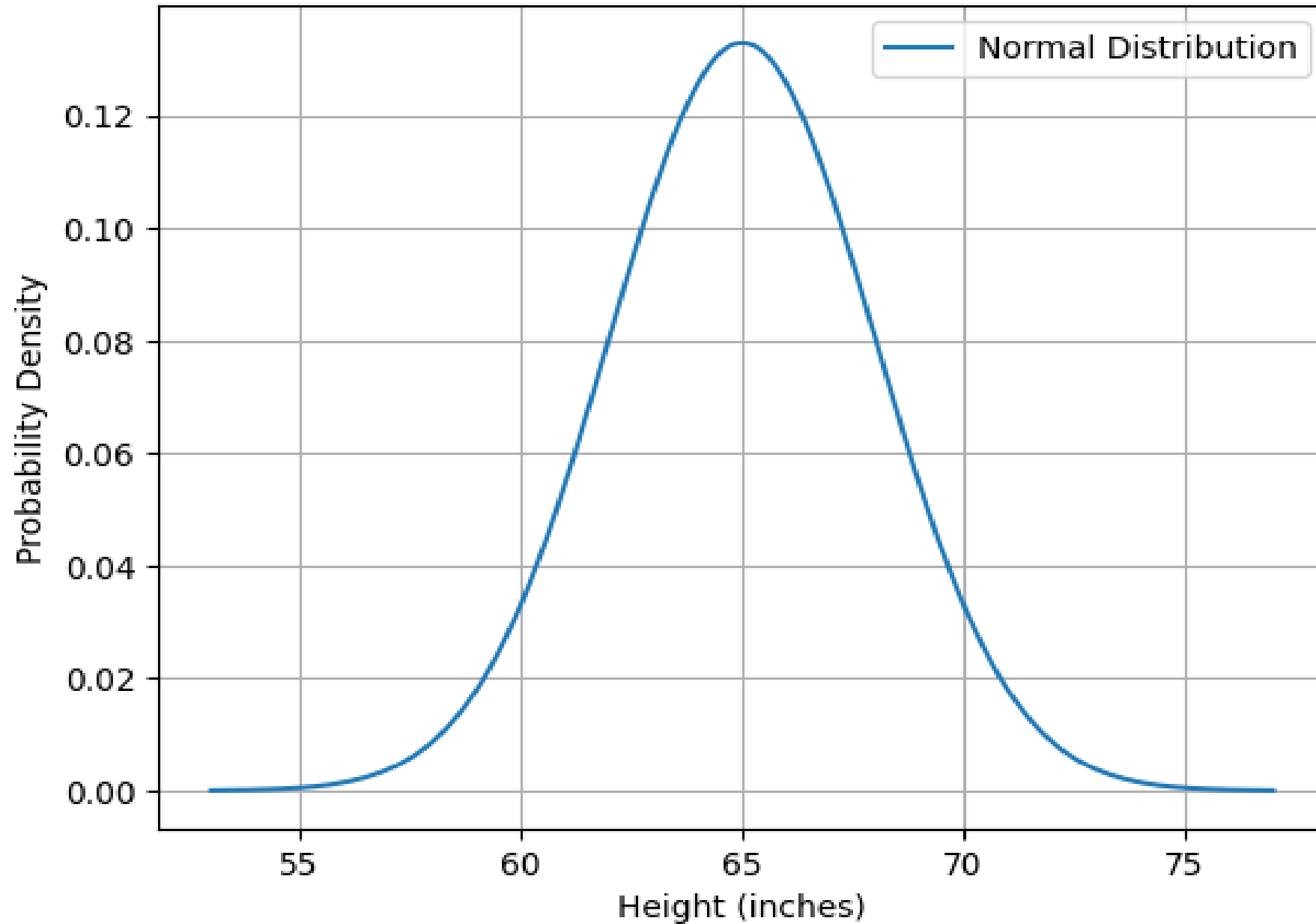
# Mean and standard deviation
mu = 65 # Mean height
sigma = 3 # Standard deviation

# Generate a range of heights (x-axis)
x = np.linspace(mu - 4*sigma, mu + 4*sigma, 1000)

# Calculate the probability density function (PDF) for each height value
pdf = stats.norm.pdf(x, mu, sigma)

# Plot the normal distribution
plt.plot(x, pdf, label='Normal Distribution')
plt.title('Normal Distribution of Heights')
plt.xlabel('Height (inches)')
plt.ylabel('Probability Density')
plt.grid(True)
plt.legend()
plt.show()
```

# Normal Distribution of Heights

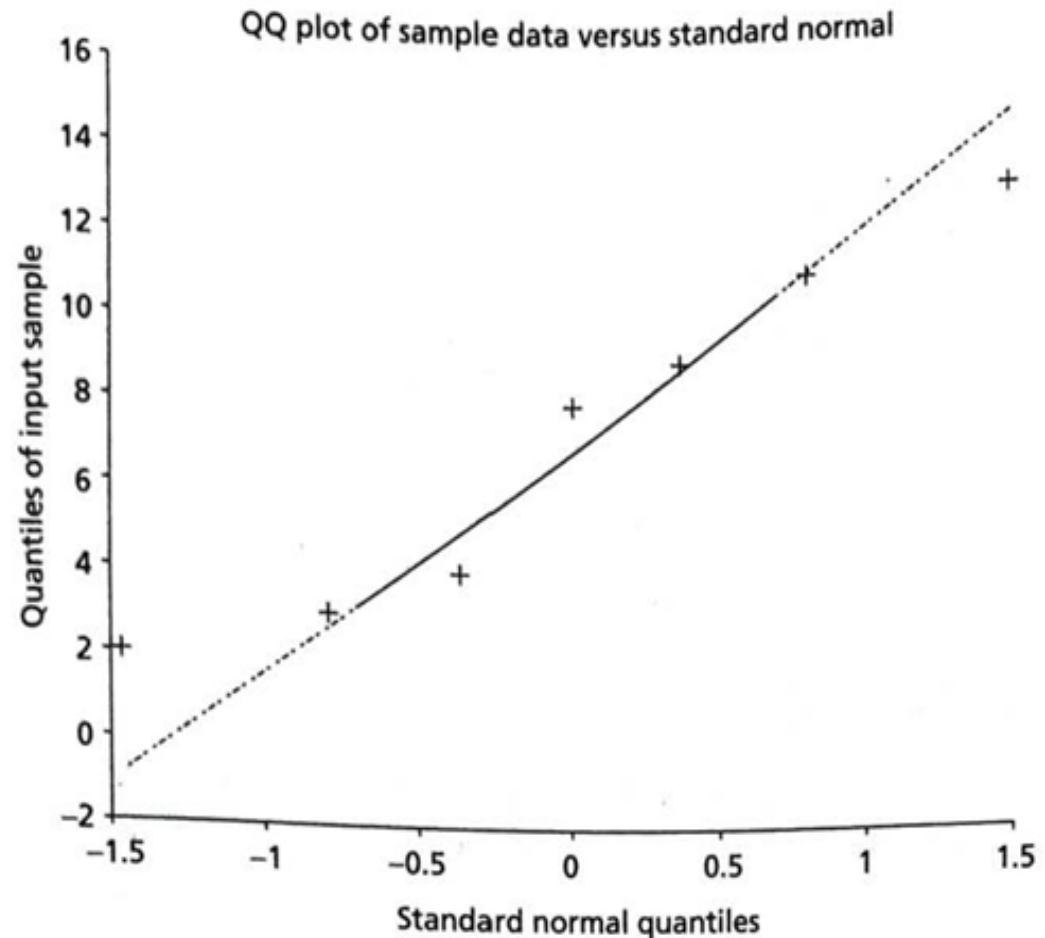


# Applications of Normal Distribution:

- **Height and Weight:** The distribution of heights and weights in a population often approximates a normal distribution.
- **Test Scores:** Scores on standardized tests often follow a normal distribution, with most students scoring around the average and fewer students scoring at the extremes.
- **Measurement Errors:** In many scientific experiments, the errors in measurements often follow a normal distribution.

# Q-Q Plot (Quantile-Quantile Plot)

- A **Q-Q Plot** is another visualization tool used to assess the **distribution shape** of a dataset. It helps determine whether the dataset follows a **normal distribution** by comparing **quantiles**.
- A **normal Q-Q plot** for the dataset **{13, 11, 2, 3, 4, 8, 9}** is shown in **Figure 2.10**.



# Quantile-Quantile (Q-Q) plot

- A **Quantile-Quantile (Q-Q) plot** is a graphical tool used to compare the distributions of two datasets by plotting their quantiles against each other. It is commonly used to assess if a dataset follows a particular theoretical distribution, such as a normal distribution.

## **Purpose:**

- The Q-Q plot helps to check the **goodness of fit** of a dataset to a specific distribution.
- It provides insight into whether two datasets come from the same distribution.
- It can help identify **skewness, kurtosis, or other deviations** from the expected distribution.

# Normal Q-Q Plot

- A **Q-Q (Quantile-Quantile) Plot** is used to **assess whether a dataset follows a normal distribution**. It compares the quantiles of the dataset against the expected quantiles of a standard normal distribution.
- **Interpretation of the Q-Q Plot**
  - **If the points align closely with the reference line (45-degree line)** → The data follows a normal distribution.
  - **If the points deviate significantly from the line** → The dataset may follow a different distribution, requiring further analysis.
- When deviations are large, it suggests that the dataset does **not** follow a normal distribution. In such cases, additional statistical investigations should be conducted before drawing conclusions.

# How It Works:

1. Quantiles of the datasets are calculated (e.g., percentiles).
2. These quantiles from the first dataset are plotted on the x-axis, and the corresponding quantiles from the second dataset (or the theoretical distribution) are plotted on the y-axis.
3. If the points lie approximately along a straight line, it indicates that the data follow the theoretical distribution.

# Example 1: Q-Q plot comparing data to a normal distribution

- Let's say we have a dataset of 100 random numbers generated from a normal distribution. We can create a Q-Q plot to compare the data to the normal distribution.
- **Steps:**
  1. **Generate Data:** We create a dataset with values from a normal distribution.
  2. **Theoretical Quantiles:** For the Q-Q plot, we compare the sample data against the quantiles of a normal distribution (theoretical quantiles).
  3. **Plot:** The points in the plot represent the quantiles of the data and the normal distribution. If the data are normally distributed, the points will lie on a straight line.

# Python Code Example:

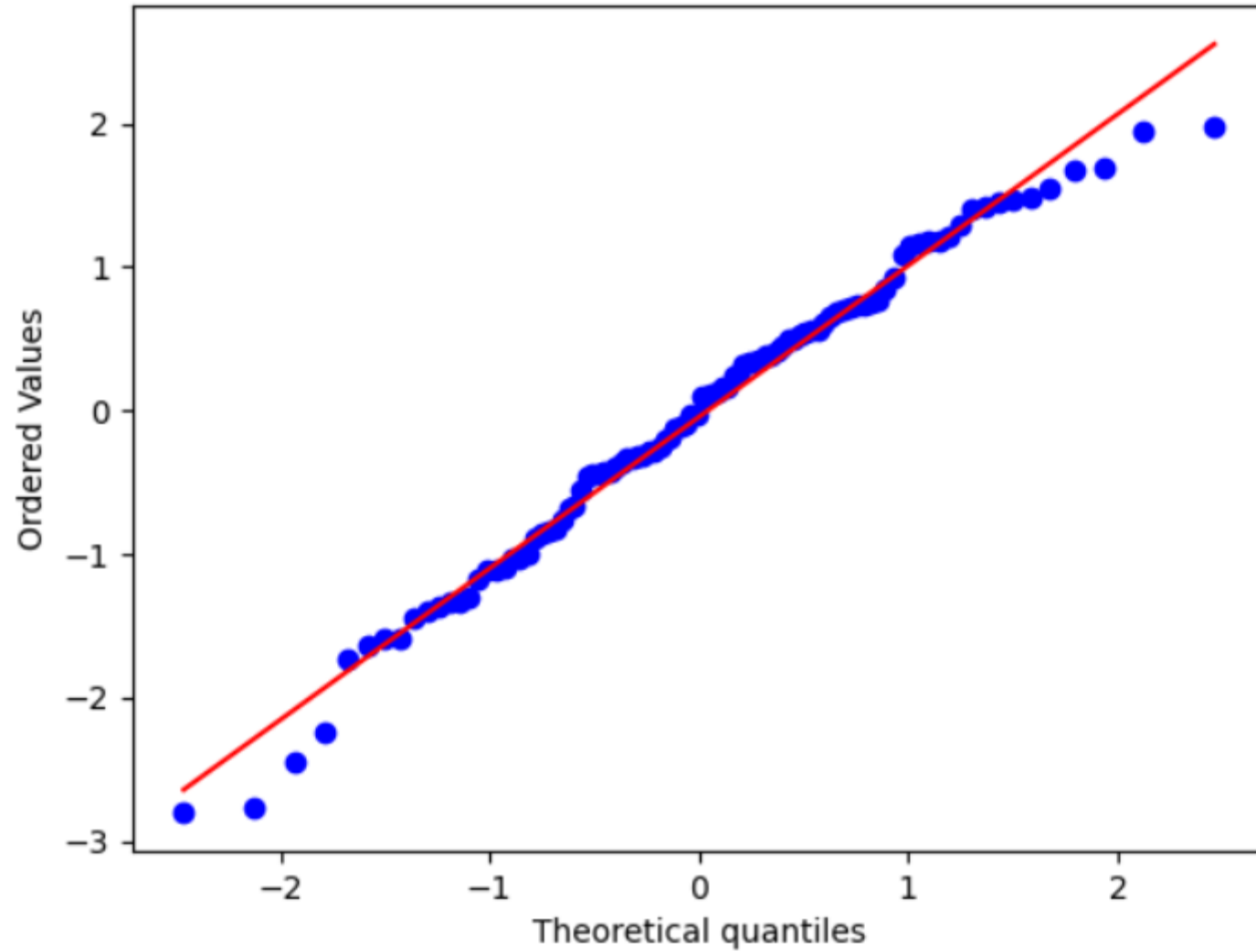
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats

# Generate a random sample from a normal distribution
data = np.random.normal(loc=0, scale=1, size=100)

# Create a Q-Q plot comparing data to a normal distribution
stats.probplot(data, dist="norm", plot=plt)

plt.title("Q-Q Plot: Normal Distribution")
plt.show()
```

Q-Q Plot: Normal Distribution



# Interpretation:

- If the points in the Q-Q plot follow a straight line, the data follow a normal distribution.
- If the points deviate from the straight line, it suggests that the data do not follow a normal distribution (e.g., heavy tails, skewness, etc.).

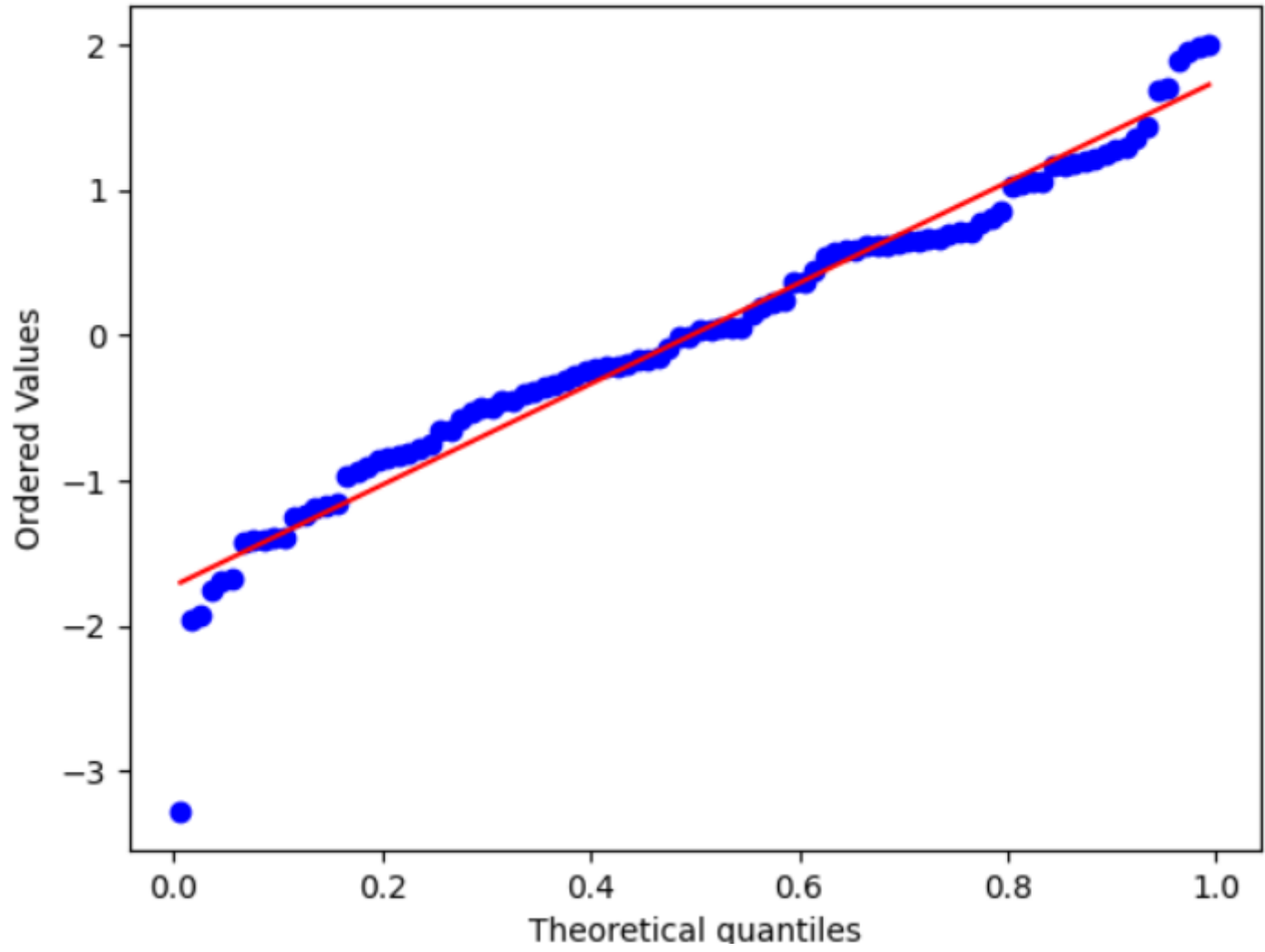
# Example 2: Q-Q plot comparing two datasets

- You can also compare two different datasets (say, data from two different experiments) using a Q-Q plot.
- **Steps:**
- **Generate two random samples** (e.g., one from a normal distribution and the other from a uniform distribution).
- **Plot** the quantiles of the two datasets against each other.

# Python Code Example

```
# Generate two random samples from different distributions  
data1 = np.random.normal(loc=0, scale=1, size=100) # Normal data  
data2 = np.random.uniform(low=0, high=1, size=100) # Uniform data  
  
# Create a Q-Q plot comparing the two datasets  
stats.probplot(data1, dist="uniform", plot=plt)  
plt.title("Q-Q Plot: Normal vs Uniform Distribution")  
plt.show()
```

Q-Q Plot: Normal vs Uniform Distribution



# Example of a Q-Q Plot Showing Deviation from Normal Distribution

- In this example, we'll:
- Generate a random dataset from a **uniform distribution**, which is clearly not normal.
- Create a Q-Q plot to compare the data to a normal distribution.
- Observe how the points deviate from the straight line, indicating that the data does not follow a normal distribution.

# Python code

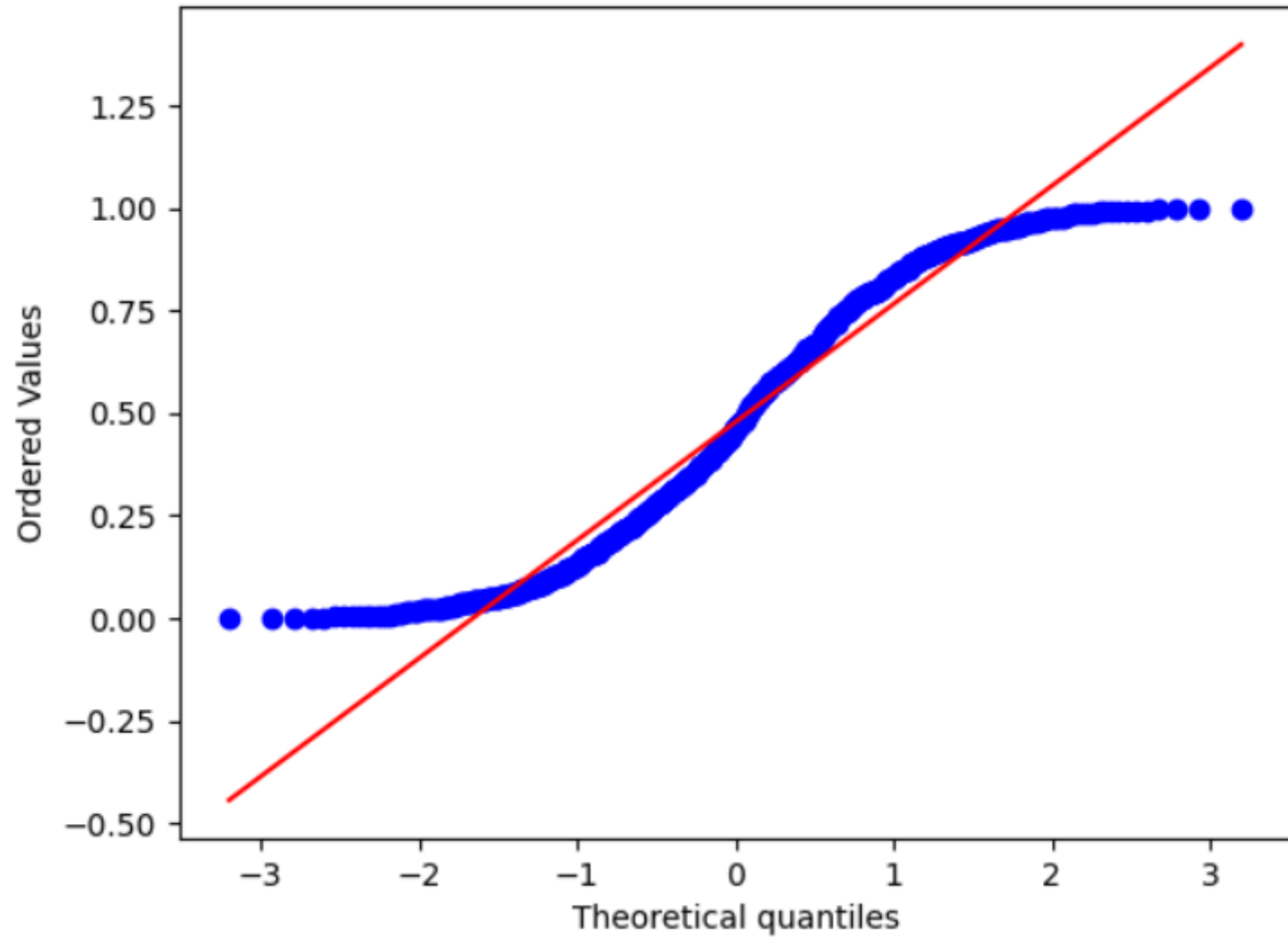
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats

# Generate a random sample from a uniform distribution (not normal)
data = np.random.uniform(low=0, high=1, size=1000)

# Create a Q-Q plot comparing the uniform distribution data to a normal distribution
stats.probplot(data, dist="norm", plot=plt)

# Display the plot
plt.title("Q-Q Plot: Uniform vs Normal Distribution")
plt.show()
```

Q-Q Plot: Uniform vs Normal Distribution



# Statistical Measures for Assessing Univariate Data

- **Skewness** → Measures asymmetry in data distribution.
- **Kurtosis** → Identifies whether data has heavy or light tails.
- **Mean Absolute Deviation (MAD)** → Measures dispersion while being robust to outliers.
- **Coefficient of Variation (CV)** → Compares variability across datasets with different scales.

# Types of Processing

1. Cloud Computing
2. Grid Computing
3. High-Performance Computing (HPC)

# Cloud Computing

- Cloud computing is a widely used **business service model** that provides scalable computing resources over the **Internet**. It is often based on a **pay-as-you-go** model and offers three primary service types:
  - 1.SaaS (Software as a Service)** – Allows users to access software applications via the cloud without installation.
  - 2.PaaS (Platform as a Service)** – Provides an environment for developers to build and deploy applications.
  - 3.IaaS (Infrastructure as a Service)** – Supplies **computing infrastructure**, such as virtual machines, storage, and networks.

# Cloud Deployment Models

Cloud computing can be deployed using the following models:

- **Public Cloud** – Accessible to the public, owned by a third-party provider.
- **Private Cloud** – Dedicated to a single organization, ensuring higher security.
- **Community Cloud** – Shared among multiple organizations with common requirements.
- **Hybrid Cloud** – A combination of **two or more cloud types** to enhance flexibility

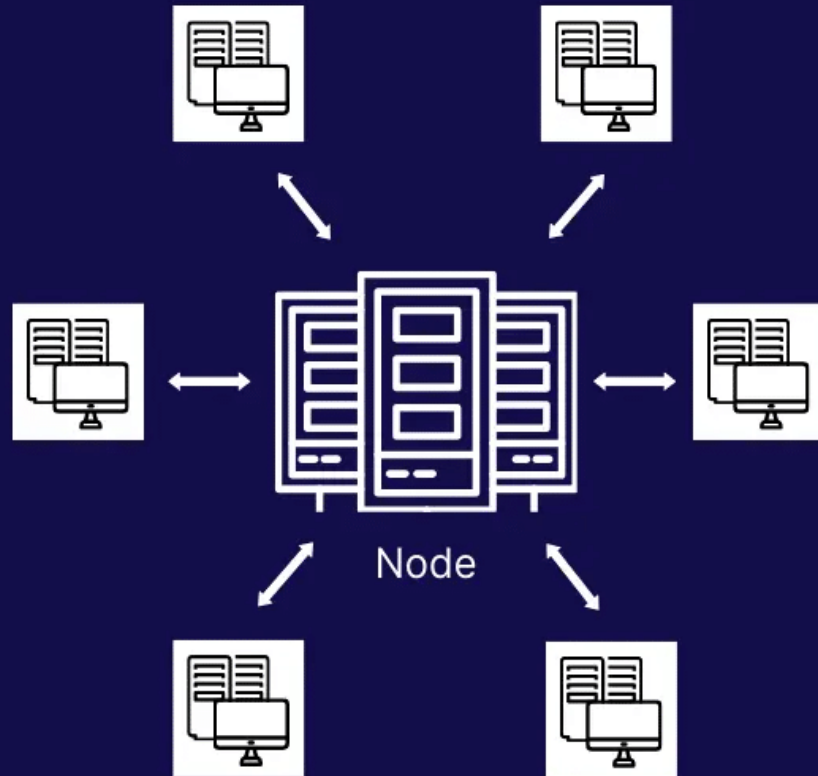
# Key Features of Cloud Computing:

- **Shared Infrastructure** – Computing resources (storage, networks) are shared.
- **Dynamic Provisioning** – Resources are assigned as needed.
- **Scalability** – Expands computing power based on demand.
- **Network Access** – Requires internet connectivity.
- **Utility-Based Metering** – Tracks usage for billing purposes.
- **Multi-Tenancy** – Supports multiple users on the same infrastructure.
- **Reliability** – Ensures availability of services.

# Grid Computing

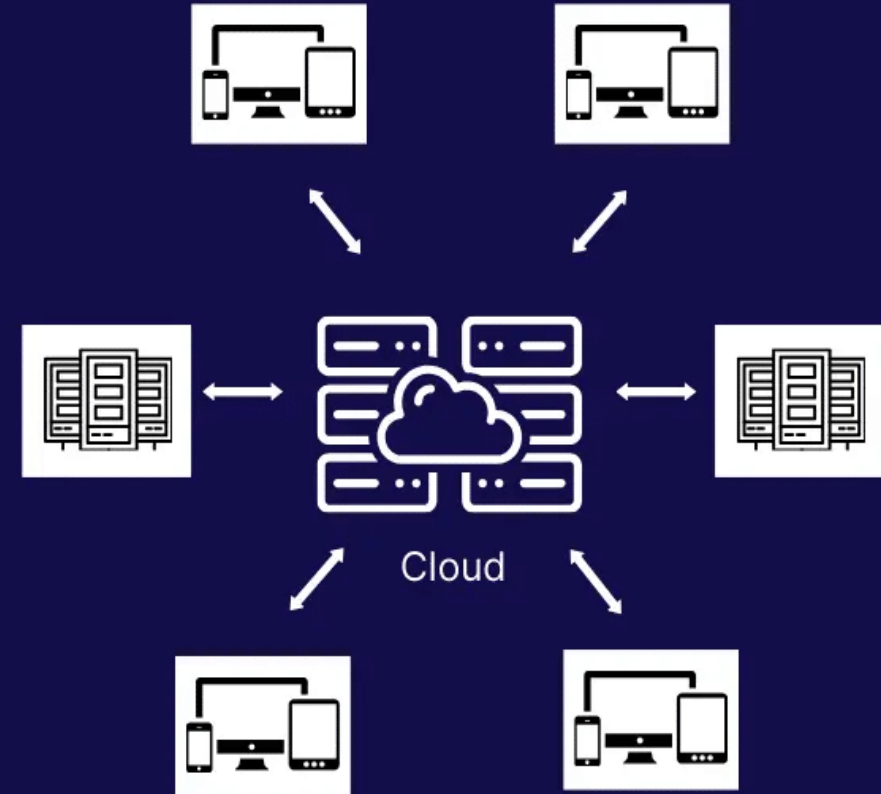
- Grid computing is a **parallel and distributed** computing framework where a network of computers functions as a **virtual supercomputer**. It is designed for **high-performance computing** (HPC) tasks that require immense processing power.
- Grid computing connects **thousands of nodes** in a **cluster**, sharing resources efficiently.
- Middleware distributes computing tasks **across multiple nodes**, allowing them to run independently before merging the results.
- Best suited for applications requiring **massive parallel computing**.

## Grid Computing



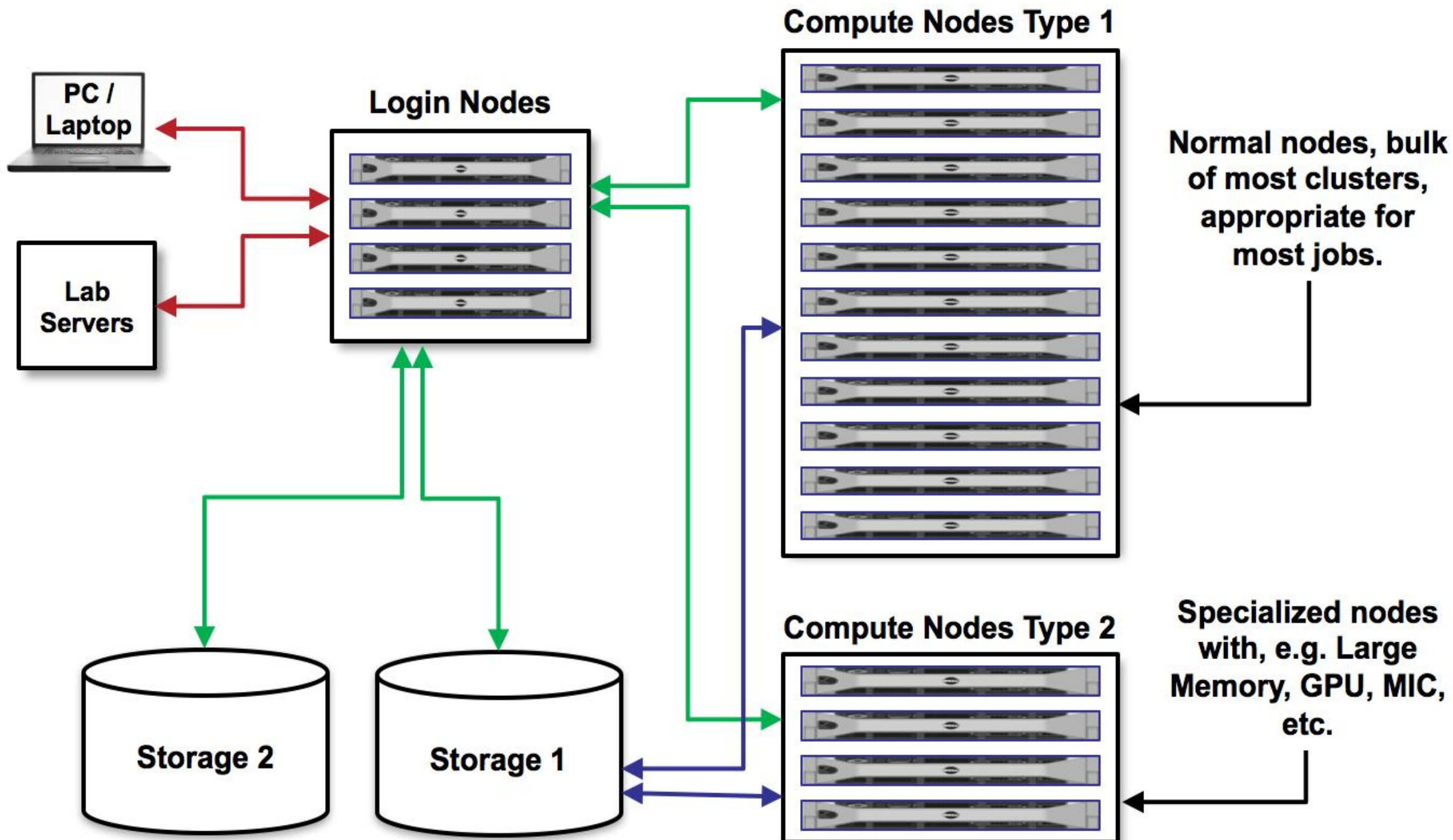
VS

## Cloud Computing



# High-Performance Computing (HPC)

- HPC involves **solving complex computational problems** at high speed by aggregating computing resources.
- Used in **scientific research, engineering, and business applications**.
- Utilizes **parallel processing techniques** for **large-scale problem-solving**.
- An **HPC system** consists of thousands of interconnected servers, divided into:
  - **Compute Nodes** – Perform tasks in parallel.
  - **Storage Nodes** – Manage data storage.
  - **Networking System** – Connects different components for efficient communication.
- HPC provides **sustained high performance**, making it essential for **data-intensive applications** such as **weather forecasting, genomic research, and financial modeling**.



# End of Module1