
Introduction to Data Mining(Unit-1)

Data mining is the process of discovering useful patterns, trends, and relationships in large datasets. It combines techniques from fields like statistics, machine learning, and database management to analyze data efficiently.

Types of Data

1. **Structured Data:**
 - Stored in relational databases, spreadsheets, or data warehouses.
 - Examples: Tables with rows (objects) and columns (attributes).
 2. **Unstructured Data:**
 - Lacks a predefined format or structure.
 - Examples: Text documents, images, audio, and videos.
 3. **Semi-Structured Data:**
 - Data with an underlying structure that is not as rigid as structured data.
 - Examples: XML files, JSON data.
 4. **Data by Context:**
 - **Transactional Data:** Captures a set of items purchased in a single transaction.
 - Example: Market basket data.
 - **Time-Series Data:** Observations measured at successive points in time.
 - Example: Stock market trends.
 - **Spatial Data:** Data related to the location of objects.
 - Example: Geographical coordinates.
 5. **Types by Attribute:**
 - **Categorical (Qualitative):** Attributes with distinct categories.
 - **Nominal:** No inherent order (e.g., gender, ID numbers).
 - **Ordinal:** Inherent order but no fixed interval (e.g., rankings).
 - **Numerical (Quantitative):** Attributes represented by numbers.
 - **Interval:** Equal intervals between values, but no true zero (e.g., temperature in Celsius).
 - **Ratio:** Equal intervals with a true zero (e.g., weight, height).
-

Data Quality

Poor data quality can lead to unreliable results in data mining. The main issues include:

1. **Noise and Outliers:**
 - Noise: Random variations or errors in data.

- Outliers: Data points that deviate significantly from the norm.
 - 2. **Missing Values:**
 - Occur due to data collection errors or unavailability.
 - Handling Methods:
 - Ignore missing data.
 - Fill with mean, median, or mode.
 - Predict missing values using machine learning techniques.
 - 3. **Duplicate Data:**
 - Redundant records caused by errors in data entry or merging datasets.
 - 4. **Inconsistency:**
 - Conflicts in data representation.
 - Example: Different date formats (e.g., MM/DD/YYYY vs. DD-MM-YYYY).
 - 5. **Bias and Representativeness:**
 - Data that is not representative of the population being studied may lead to skewed results.
-

Data Processing (Preprocessing)

Data preprocessing is a crucial step to prepare raw data for analysis.

1. **Data Cleaning:**
 - Detect and correct missing, noisy, or inconsistent data.
 2. **Data Integration:**
 - Combine data from multiple sources into a coherent dataset.
 3. **Data Transformation:**
 - Convert data into a format suitable for mining.
 - Methods include:
 - **Normalization:** Scale numerical values to a specific range (e.g., [0, 1]).
 - **Discretization:** Convert continuous data into categorical bins.
 - **Encoding:** Convert categorical data into numerical format (e.g., one-hot encoding).
 4. **Data Reduction:**
 - Reduce dataset size while preserving its essential information.
 - Techniques include:
 - Dimensionality reduction (e.g., PCA).
 - Sampling.
 - Aggregation.
 5. **Data Subsetting:**
 - Select relevant attributes or objects based on criteria.
-

Measures of Similarity and Dissimilarity

These measures are used to compare data objects and are critical for tasks like clustering, classification, and anomaly detection.

Similarity Measures

1. Cosine Similarity:

- Measures the cosine of the angle between two vectors.
- Formula:

1. Cosine Similarity:

- Measures the cosine of the angle between two vectors.
- Formula:

$$\text{Similarity} = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \|\vec{B}\|}$$

- Example: Text analysis (comparing document vectors).

-
- Example: Text analysis (comparing document vectors).

2. Jaccard Similarity:

- Measures similarity between sets.

2. Jaccard Similarity:

- Measures similarity between sets.
- Formula:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

- Example: Comparing two users' preferences.

-
- Example: Comparing two users' preferences.

Dissimilarity Measures

1. Euclidean Distance:

- The straight-line distance between two points in a multidimensional space.
-
- Example: Measuring spatial distances.

1. Euclidean Distance:

- The straight-line distance between two points in a multidimensional space.
- Formula:

$$d(A, B) = \sqrt{\sum_{i=1}^n (A_i - B_i)^2}$$

- Example: Measuring spatial distances.

2. Manhattan Distance:

- The sum of absolute differences between corresponding attributes.
- Formula:

$$d(A, B) = \sum_{i=1}^n |A_i - B_i|$$

- Example: Delivery route optimization.

3. Minkowski Distance:

- Generalized form of Euclidean and Manhattan distances.
- Formula:

$$d(A, B) = \left(\sum_{i=1}^n |A_i - B_i|^p \right)^{1/p}$$

- For $p = 2$, it becomes Euclidean distance; for $p = 1$, it becomes Manhattan distance.

4. Hamming Distance:

- Measures the number of differing positions between two strings of equal length.
- Example: Error detection in coding.

Properties of Good Measures

Properties of Good Measures

- Symmetry: $d(A, B) = d(B, A)$.
- Non-negativity: $d(A, B) \geq 0$.
- Identity: $d(A, B) = 0$ if and only if $A = B$.
- Triangle Inequality: $d(A, B) + d(B, C) \geq d(A, C)$.

Exploring Data

Exploring data involves examining datasets to understand their structure, characteristics, and relationships between variables. This step is essential for identifying patterns, trends, and anomalies, and it lays the foundation for effective data analysis.

1. Dataset

Definition:

A dataset is a collection of data points organized in a structured format, often as a table.

- **Rows:** Represent individual records or objects (e.g., customers, products).
- **Columns:** Represent attributes or features of the objects (e.g., age, income).

Types of Datasets:

1. **Transactional Data:**
 - Represents individual transactions.
 - Example: Purchase records from a store.
 2. **Spatial Data:**
 - Contains information about the location of objects.
 - Example: Satellite images, GIS data.
 3. **Time-Series Data:**
 - Observations are made over time.
 - Example: Daily stock prices.
 4. **Text Data:**
 - Unstructured or semi-structured textual information.
 - Example: Social media posts.
-

2. Summary Statistics

Summary statistics provide a quick overview of the dataset, describing its main features quantitatively.

Measures of Central Tendency:

1. **Mean:** The average of all values.

Formula:

$$\text{Mean} = \frac{\sum_{i=1}^n x_i}{n}$$

2. **Median:** The middle value when data is ordered.
 - Useful for skewed distributions.
3. **Mode:** The most frequently occurring value.

Measures of Dispersion:

1. **Range:** The difference between the maximum and minimum values.
2. **Variance:** Measures how much values deviate from the mean.

• Formula:

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

3. **Standard Deviation:** The square root of variance, representing dispersion in the same units as the data.

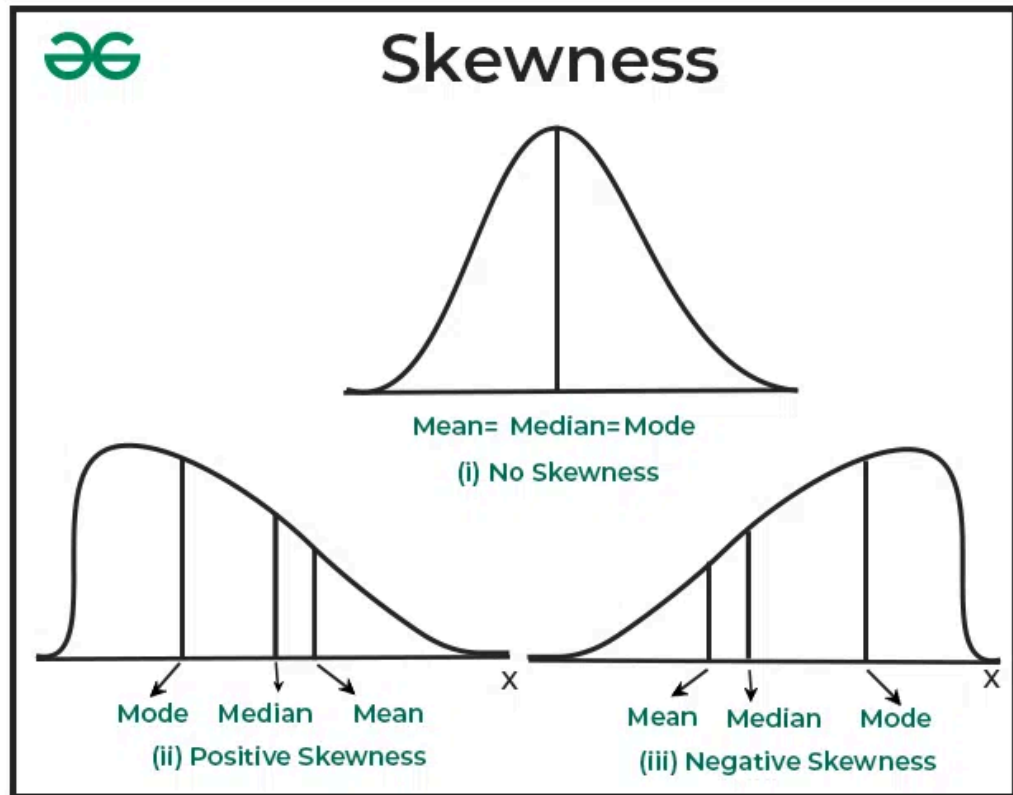
• Formula:

$$\sigma = \sqrt{\sigma^2}$$

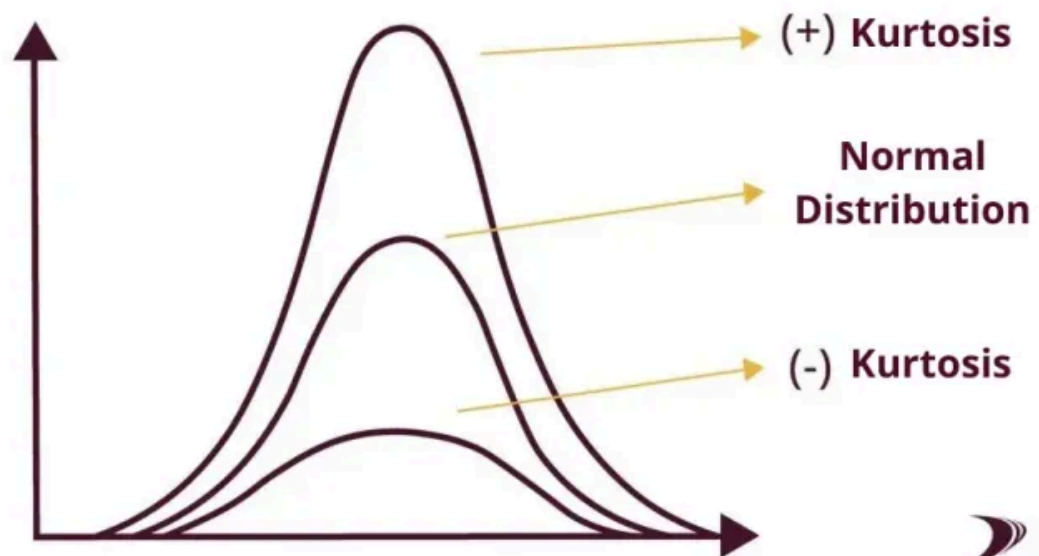
4. **Interquartile Range (IQR):** The range between the first quartile (Q1) and the third quartile (Q3).

Shape of Data Distribution:

1. **Skewness:** Measures asymmetry of the distribution.
 - Positive skew: Tail on the right.
 - Negative skew: Tail on the left.



2. **Kurtosis:** Measures the "tailedness" of the distribution.



3. Visualization

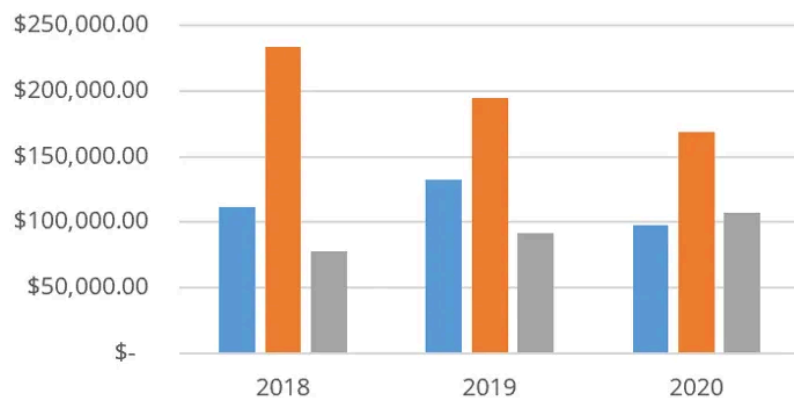
Visualization helps represent data graphically, making it easier to identify patterns, trends, and outliers.

Common Visualization Techniques:

1. Bar Charts:

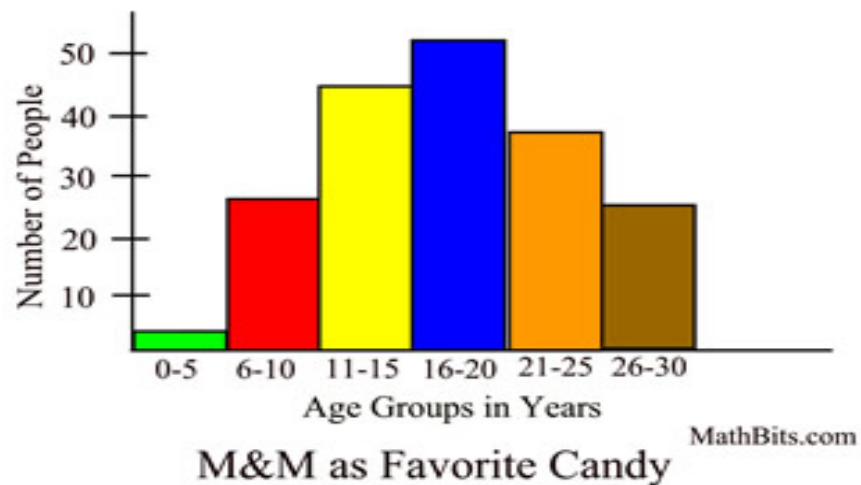
- Visualize categorical data using bars.
- Example: Sales by product category.

Bar Chart (Data Visualization)



2. Histograms:

- Show the distribution of numerical data.
- Example: Frequency of customer ages.



3. Scatter Plots:

- Represent relationships between two numerical variables.

- Example: Relationship between advertising spend and sales.

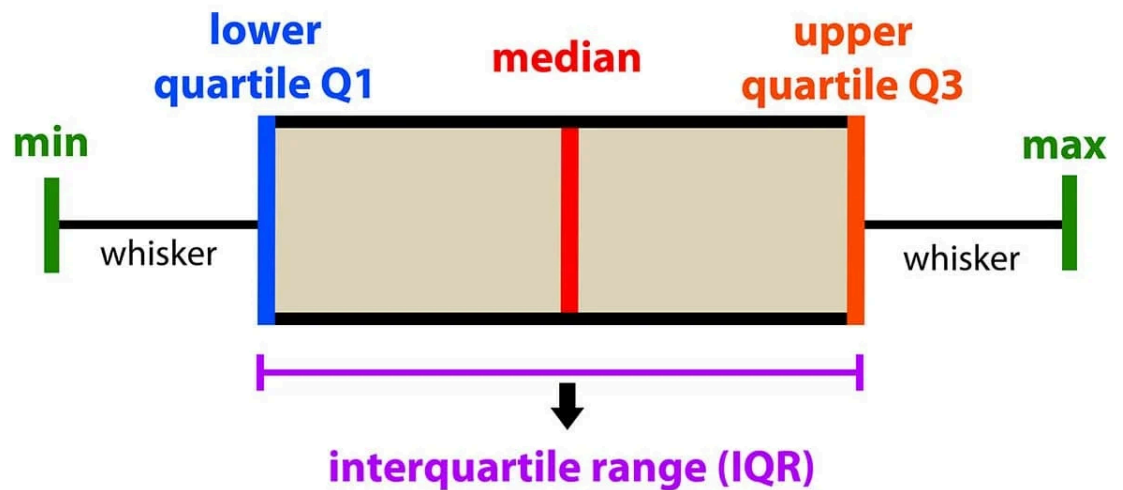


○

4. Box Plots:

- Summarize data distribution, highlighting medians, quartiles, and outliers.

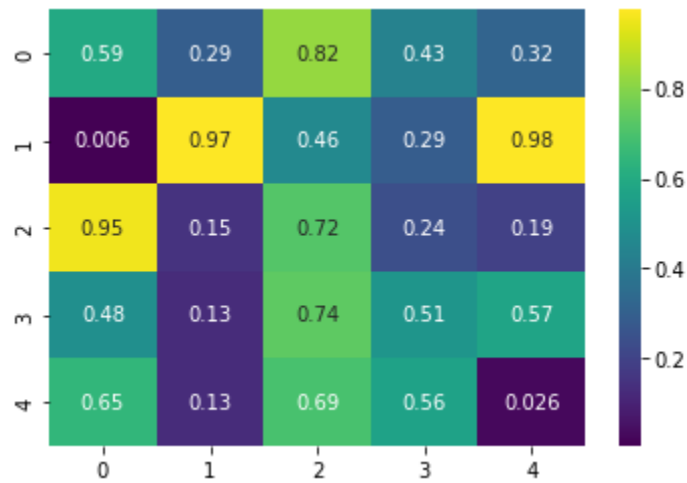
introduction to data analysis: Box Plot



○

5. Heatmaps:

- Represent data intensity using color gradients.
- Example: Correlation matrices.



6. Line Charts:

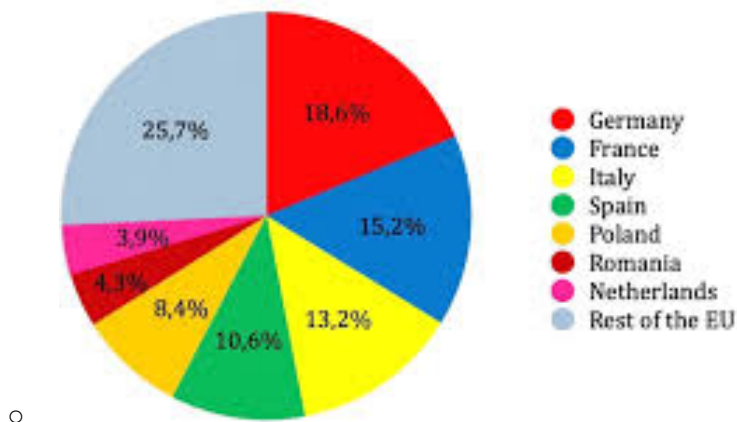
- Visualize trends over time.
- Example: Website traffic over months.



7. Pie Charts:

- Represent proportions in categorical data.
- Example: Market share distribution.

Population of Countries of the European Union
in 2021 by percentage



4. OLAP (Online Analytical Processing) and Multidimensional Data Analysis

OLAP and multidimensional analysis provide powerful tools for exploring large datasets, often in business and decision-making contexts.

OLAP:

1. Definition:

- A technology that enables fast, interactive analysis of multidimensional data.
- Often used in data warehousing and business intelligence.

2. Core Operations:

- **Roll-Up:** Aggregates data along a dimension hierarchy (e.g., summarizing sales by year from monthly data).
- **Drill-Down:** Explores detailed data by breaking down aggregates (e.g., analyzing sales by month or day).
- **Slice:** Extracts a subset of data by filtering on one dimension (e.g., sales in a specific region).
- **Dice:** Creates a subcube by filtering on multiple dimensions (e.g., sales in Region A for Product X).
- **Pivot:** Reorients the data cube to view it from different perspectives.

Multidimensional Data Analysis:

1. Concept:

- Data is organized in multiple dimensions, often visualized as a cube (data cube).
- Dimensions: Categorical attributes (e.g., time, location, product).
- Measures: Quantitative metrics (e.g., sales, revenue).

2. Example:

- Dimensions: Time, region, product.
- Measure: Sales.
- A data cube can provide sales totals across combinations of these dimensions.

Applications:

- **Retail:** Analyze sales performance by store, time, and product.
 - **Finance:** Track revenue by branch, customer segment, and period.
 - **Healthcare:** Monitor patient outcomes across hospitals and treatments.
-

Classification: Basic Concepts (Unit-2)

Definition

Classification is a supervised learning task where the goal is to assign predefined labels (categories) to input data based on its features. The model learns from labeled training data and predicts labels for new, unseen data.

Examples

1. Email spam detection: Classifying emails as "spam" or "not spam."
2. Disease diagnosis: Predicting whether a patient has a disease based on symptoms.
3. Customer segmentation: Categorizing customers into groups (e.g., premium, regular).

Key Concepts

1. **Features (Attributes):** Variables used to predict the target class (e.g., age, income).
 2. **Target Variable:** The label or class being predicted (e.g., spam, not spam).
 3. **Training Data:** Data used to train the model, including both features and target labels.
 4. **Test Data:** Data used to evaluate the performance of the trained model.
-

General Approach for Solving a Classification Problem

1. **Understand the Problem:** Define objectives, features, and target classes.
2. **Data Preprocessing:**
 - Clean and transform data.
 - Handle missing values and encode categorical variables.
3. **Feature Selection/Engineering:**
 - Select relevant features.
 - Engineer new features to improve model performance.
4. **Split Data:**

- Divide the dataset into training and test sets (e.g., 70%-30%).
 - 5. **Train Model:** Use training data to build the classifier.
 - 6. **Evaluate Model:** Assess performance using metrics like accuracy, precision, recall, etc.
 - 7. **Optimize Model:** Tweak hyperparameters and retrain to improve performance.
 - 8. **Deploy Model:** Use the trained model for real-world predictions.
-

Decision Tree Induction

Definition

A decision tree is a tree-like structure where:

- **Nodes** represent features or attributes.
- **Edges** represent decisions or conditions.
- **Leaves** represent class labels.

How It Works

1. Start with the entire dataset at the root.
2. Select the best feature to split the data (using criteria like Gini Index or Information Gain).
3. Repeat splitting recursively until:
 - All instances belong to the same class.
 - A stopping criterion is met (e.g., tree depth limit).

Example

Dataset: Predict if someone will buy a product based on age and income.

- Root node: "Age < 30?"
- Split: Yes → Low income; No → High income.

Splitting Criteria

1. **Gini Index:** Measures impurity of a node. Lower Gini indicates better splits.

$$Gini = 1 - \sum_{i=1}^C p_i^2$$

Where p_i is the proportion of instances in class i .

2. **Information Gain (IG):** Measures reduction in entropy after a split.

$$IG = Entropy_{parent} - \sum_{i=1}^k \frac{|D_i|}{|D|} \times Entropy_{child}$$

Where:

$$Entropy = - \sum_{i=1}^C p_i \log_2(p_i)$$

Model Overfitting

Overfitting occurs when the model learns not just the general patterns but also noise or irrelevant details in the training data.

Causes of Overfitting

1. **Noise in the Data:**
 - Random errors or outliers in the training data can mislead the model.
 - Example: A classifier that uses rare features specific to certain samples.
2. **Lack of Representative Samples:**
 - If the training data doesn't cover the diversity of real-world scenarios, the model fails to generalize.
 - Example: Training a model only on sunny weather conditions for a rain prediction task.

Preventing Overfitting

1. **Pruning:** Simplify decision trees by removing unnecessary branches.
2. **Regularization:** Add constraints (e.g., max tree depth, minimum samples per leaf).

3. **Cross-Validation:** Use k-fold cross-validation to test the model on multiple subsets of data.
4. **Increase Training Data:** Provide more diverse samples to improve generalization.

Evaluating the Performance of a Classifier

Metrics

Metrics

1. **Accuracy:** Proportion of correctly classified instances.

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

2. **Precision:** Proportion of true positives among predicted positives.

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

3. **Recall (Sensitivity):** Proportion of true positives among actual positives.

$$Recall = \frac{TP}{TP + FN}$$

4. **F1-Score:** Harmonic mean of precision and recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

5. **Confusion Matrix:**

- A table showing actual vs. predicted classifications.

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

1.

ROC Curve (Receiver Operating Characteristic):

- Plots the true positive rate (TPR) against the false positive rate (FPR) for different thresholds.
- **AUC (Area Under Curve):** Measures overall model performance; higher is better.

Techniques

- 1. **Train-Test Split:** Split data into training and test sets to evaluate performance on unseen data.
- 2. **Cross-Validation:** Divide data into k subsets, train on k-1, and test on the remaining subset iteratively.
- 3. **Bootstrapping:** Sample with replacement to create multiple datasets for evaluation.

Example for classification:

Let's manually calculate the key steps for classification using the Iris dataset example. I'll simulate a portion of the data to show calculations for:

- 1. Building the Decision Tree
- 2. Evaluating Metrics (Confusion Matrix, Accuracy, Precision, Recall, F1-Score)

Sample Dataset

Here's a simplified version of the Iris dataset for 6 samples and 2 classes:

Sample	Sepal Length	Sepal Width	Petal Length	Petal Width	Class (Target)
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
3	6.2	2.9	4.3	1.3	Versicolor
4	5.9	3.0	4.2	1.5	Versicolor
5	6.5	3.0	5.2	2.0	Virginica
6	7.0	3.2	4.7	1.4	Virginica

Step 1: Decision Tree Induction

Splitting Criteria (Using Gini Index)

1. Compute Gini for Root Node:

At the root, all 6 samples are considered. The distribution is:

- Setosa: 2
- Versicolor: 2
- Virginica: 2

Gini Index formula:

$$Gini = 1 - \sum_{i=1}^k (p_i)^2$$

Where p_i is the proportion of samples in class i .

$$Gini = 1 - [(2/6)^2 + (2/6)^2 + (2/6)^2] = 1 - [0.111 + 0.111 + 0.111] = 1 - 0.333 = 0.667$$

2. Split on a Feature (e.g., Petal Length ≤ 2.5):

- Left node: Samples {1, 2} (Setosa: 2).
- Right node: Samples {3, 4, 5, 6} (Versicolor: 2, Virginica: 2).

Left Node Gini:

$$Gini = 1 - [(2/2)^2] = 1 - 1 = 0$$

Right Node Gini:

$$Gini = 1 - [(2/4)^2 + (2/4)^2] = 1 - [0.25 + 0.25] = 1 - 0.5 = 0.5$$

Weighted Gini for Split:

$$Weighted\ Gini = \frac{2}{6}(0) + \frac{4}{6}(0.5) = 0 + 0.333 = 0.333$$

This split improves purity (from 0.667 to 0.333). Hence, the first decision is:

If Petal Length ≤ 2.5 , classify as Setosa. Else, split further.

Step 2: Evaluating the Performance of Classifier

Using the tree for predictions on the 6 samples, we assume predictions as follows:

- Predictions: {Setosa, Setosa, Versicolor, Versicolor, Virginica, Virginica}.
- True Labels: {Setosa, Setosa, Versicolor, Versicolor, Virginica, Virginica}.

Confusion Matrix

	Predicted Setosa	Predicted Versicolor	Predicted Virginica
Actual Setosa	2	0	0
Actual Versicolor	0	2	0
Actual Virginica	0	0	2

Accuracy Calculation

$$Accuracy = \frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{2 + 2 + 2}{6} = 1.0$$

Precision, Recall, and F1-Score

1. Precision for Setosa:

$$Precision = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{2}{2} = 1.0$$

2. Recall for Setosa:


$$Recall = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} = \frac{2}{2} = 1.0$$

3. F1-Score for Setosa:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \times \frac{1.0 \times 1.0}{1.0 + 1.0} = 1.0$$

Repeat similar calculations for Versicolor and Virginica.

Summary of Results

- Accuracy: 1.0
 - Precision, Recall, F1-Score: Perfect (1.0) for all classes because the classifier perfectly predicted all samples.
- 

=====

Classification - Alternative Techniques:

1. Nearest Neighbor Classifier (k-NN)

Concept

The k-Nearest Neighbor (k-NN) classifier is a simple, instance-based learning algorithm. It assigns a class label to a new data point based on the majority class of its k nearest neighbors in the feature space.

How it Works

1. Calculate the distance between the test data point and all training data points.

How it Works

1. Calculate the distance between the test data point and all training data points.

- Common distance metrics:

- Euclidean Distance:

$$d(A, B) = \sqrt{\sum_{i=1}^n (A_i - B_i)^2}$$

- Manhattan Distance:

$$d(A, B) = \sum_{i=1}^n |A_i - B_i|$$

2. Identify the k nearest neighbors.

3. Assign the majority class among these neighbors to the test data point.

2. Identify the k nearest neighbors.

3. Assign the majority class among these neighbors to the test data point.

Advantages

- Simple to implement.
- Non-parametric (no assumption about data distribution).

Disadvantages

- Computationally expensive, especially for large datasets.
- Sensitive to the choice of k and distance metric.

2. Bayesian Classifier (Naive Bayes)

Concept

The Bayesian Classifier uses Bayes' Theorem to predict the probability that a data point belongs to a certain class. The "naive" assumption is that features are conditionally independent given the class label.

Bayes' Theorem:

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

Where:

- $P(C|X)$: Posterior probability of class C given the data X .
- $P(X|C)$: Likelihood of data X given class C .
- $P(C)$: Prior probability of class C .
- $P(X)$: Evidence (normalizing constant).

Steps:

1. Compute the prior probabilities $P(C)$ for each class.
2. Calculate $P(X|C)$ for each class by assuming feature independence:

$$P(X|C) = P(x_1|C) \cdot P(x_2|C) \cdot \dots \cdot P(x_n|C)$$

3. Use Bayes' Theorem to compute posterior probabilities and assign the class with the highest posterior.

Advantages

- Works well for high-dimensional data.
- Simple and computationally efficient.

Disadvantages

- Relies on the assumption of feature independence, which may not hold in practice.

3. Support Vector Machines (SVM)

Concept

SVMs are supervised learning algorithms that find a hyperplane to separate data points into classes. The objective is to maximize the margin (distance) between the hyperplane and the closest data points (support vectors) from each class.

Linear SVM (Separable Case)

1. Definition:

- When the data is linearly separable, SVM identifies a straight line (or hyperplane in higher dimensions) that perfectly separates the classes.

2. Mathematical Formulation:

- Hyperplane equation:

$$w \cdot x + b = 0$$

Where w is the weight vector, x is the data point, and b is the bias.

- The margin is maximized subject to:

$$y_i(w \cdot x_i + b) \geq 1, \quad \forall i$$

Where y_i is the class label (+1 or -1).

3. Optimization Problem:

- Minimize:

$$\frac{1}{2} \|w\|^2$$

Subject to the constraints above.

○

Non-Separable Case (Soft Margin SVM)

1.

1. Definition:

- In real-world scenarios, data is often not perfectly separable. SVM introduces slack variables (ξ_i) to allow for some misclassifications.

2. Modified Constraints:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

3. Optimization Problem:

- Minimize:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

Where C is a regularization parameter balancing margin maximization and misclassification.

=====

(Another concept)

Kernel Trick

For non-linear data, SVM uses kernel functions to transform data into a higher-dimensional space where it becomes linearly separable.

- Common Kernels:
 1. Linear Kernel: $K(x_i, x_j) = x_i \cdot x_j$.
 2. Polynomial Kernel: $K(x_i, x_j) = (x_i \cdot x_j + c)^d$.
 3. RBF Kernel (Gaussian): $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$.

Comparison of Techniques

Technique	Advantages	Disadvantages
Nearest Neighbor (k-NN)	Simple, non-parametric	Computationally expensive for large datasets
Bayesian Classifier	Works well with high-dimensional data	Assumes feature independence
SVM	Effective for high-dimensional data, works with non-linear boundaries (kernel trick)	Can be computationally expensive, sensitive to parameter tuning



Association Analysis(Unit-3)

Problem Definition

Association analysis aims to discover interesting relationships, patterns, or associations among items in transactional datasets.

- **Example:** Market Basket Analysis, which identifies products frequently bought together.

Key Terms

1. **Itemset:** A collection of items (e.g., {milk, bread}).
2. **Frequent Itemset:** An itemset that occurs frequently in the dataset, based on a minimum support threshold.

3. **Support:** Fraction of transactions containing an itemset.

$$\text{Support}(X) = \frac{\text{Transactions containing } X}{\text{Total Transactions}}$$

4. **Confidence:** Likelihood that an item Y is purchased given that X is purchased.

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

5. **Lift:** Measures the strength of a rule relative to random chance.

$$\text{Lift}(X \rightarrow Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X) \times \text{Support}(Y)}$$

3.
 - Lift > 1 indicates a positive association between X and Y .

Frequent Itemset Generation

Objective

Identify all itemsets that meet a minimum support threshold.

Apriori Algorithm

1. **Key Idea:** Uses the downward closure property (if an itemset is frequent, all its subsets are also frequent).
2. **Steps:**
 - Generate candidate itemsets of length $k+1$.
 - Count the support of these candidates.
 - Prune itemsets that do not meet the minimum support threshold.
 - Repeat for $k+1$ until no more candidates can be generated.

Rule Generation

Once frequent itemsets are identified, rules are generated to find interesting patterns.

- **Example:** From the frequent itemset {milk,bread,butter}\{milk, bread, butter}\{milk,bread,butter}, generate rules like:
 - Milk,bread→butter
 - milk→bread,butter

Rules must satisfy the minimum confidence threshold.

Compact Representation of Frequent Itemsets

Challenges

Frequent itemsets may overlap, leading to redundant rules. Compact representations aim to reduce redundancy.

Techniques:

Techniques:

1. Maximal Frequent Itemsets:

- Largest itemsets where no supersets are frequent.
- Example: If $\{A, B, C\}$ is frequent, subsets like $\{A, B\}$ are not explicitly stored.

2. Closed Frequent Itemsets:

- Itemsets where no superset has the same support.
- Example: If $\{A, B, C\}$ and $\{A, B\}$ have the same support, only $\{A, B, C\}$ is stored.

1.

FP-Growth Algorithm

Overview

An efficient alternative to Apriori, FP-Growth avoids candidate generation by using a compact data structure called the **FP-tree (Frequent Pattern Tree)**.

Steps:

1. **Construct FP-Tree:**

- Organize transactions into a tree structure, grouping shared prefixes.
- Record item frequencies along paths.
- 2. **Mine FP-Tree:**
 - Extract frequent itemsets by recursively growing item combinations from the tree.

Advantages:

- Reduces computational overhead by avoiding repeated scans of the dataset.
 - Handles large datasets effectively.
-

Handling Categorical and Continuous Attributes

1. **Categorical Attributes:**
 - Directly used as items in transactions.
 - Example: Gender = Male becomes an item in the dataset.
 2. **Continuous Attributes:**
 - Discretization: Divide continuous attributes into intervals or bins.
 - Example: Age (20–30, 31–40, etc.).
-

Concept Hierarchy

Definition

A concept hierarchy organizes data attributes into levels of granularity or abstraction.

- **Example:**
 - Geographic hierarchy: Country → State → City.
 - Product hierarchy: Electronics → Laptops → Gaming Laptops.

Applications:

1. **Generalization:** Analyze patterns at higher levels (e.g., country-level sales).
 2. **Specialization:** Drill down into finer details (e.g., city-level trends).
-

Sequential Patterns

Definition

Identifying frequent subsequences in datasets where order matters.

- **Example:**
 - Transactional data: $A \rightarrow B \rightarrow C$.
 - A sequential pattern could be: $A \rightarrow C$.

Applications:

- Customer purchase behavior.
 - Web clickstream analysis.
-

Subgraph Patterns

Definition

Finding frequent subgraphs in graph-structured data.

- **Example:** Social networks (e.g., common friend groups), molecular structures (e.g., chemical substructures).

Steps:

1. Represent data as a graph.
2. Apply algorithms like Apriori-based or pattern growth methods to identify frequent subgraphs.

Applications:

- Analyzing social connections.
- Identifying frequent molecular patterns in drug discovery.

=====

Clustering: Overview(Unit-4)

Definition

Clustering is an unsupervised learning technique that groups data points into clusters based on their similarity. Data points within a cluster are more similar to each other than to points in other clusters.

Applications

1. Customer segmentation in marketing.
2. Document clustering for information retrieval.
3. Image segmentation in computer vision.
4. Anomaly detection in fraud detection.

Types of Clustering

1. **Partitioning Methods:** Divides data into non-overlapping subsets (e.g., k-means).
 2. **Hierarchical Methods:** Builds a tree-like structure of clusters (e.g., agglomerative clustering).
 3. **Density-Based Methods:** Groups points based on density (e.g., DBSCAN).
 4. **Model-Based Methods:** Assumes a statistical distribution for clusters (e.g., Gaussian Mixture Models).
-

K-Means Clustering

Concept

K-Means is a partitioning method that divides n data points into k clusters by minimizing the variance within each cluster.

Steps:

1. Initialize k cluster centroids randomly.
2. Assign each data point to the nearest centroid based on a distance metric (e.g., Euclidean distance).
3. Update centroids by computing the mean of all points in each cluster.
4. Repeat steps 2 and 3 until convergence (no change in assignments or centroids).

Advantages:

- Simple and efficient for large datasets.
- Works well for spherical clusters.

Disadvantages:

- Sensitive to initial centroid placement.
 - Struggles with non-spherical clusters or varying cluster sizes.
-

Agglomerative Hierarchical Clustering

Concept

Agglomerative clustering starts with each data point as a separate cluster and merges the closest clusters iteratively until a single cluster is formed or a stopping criterion is met.

Steps:

1. Compute a proximity matrix (e.g., distances between points).
2. Identify the two closest clusters and merge them.
3. Update the proximity matrix to reflect the new cluster.
4. Repeat until all data points are in a single cluster.

Linkage Methods:

1. **Single Linkage:** Minimum distance between points in two clusters.
2. **Complete Linkage:** Maximum distance between points in two clusters.
3. **Average Linkage:** Average distance between all points in two clusters.

Advantages:

- Captures nested clustering structures.
- Does not require the number of clusters as input.

Disadvantages:

- Computationally expensive for large datasets.
 - Sensitive to noisy data and outliers.
-

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

Concept

DBSCAN groups points based on their density and identifies outliers as noise.

Key Parameters:

1. **epsilon(ϵ)**: Maximum radius of the neighborhood for a point to be considered in a cluster.
2. **MinPts**: Minimum number of points required to form a dense region.

Steps:

1. Label all points as **core**, **border**, or **noise**:
 - **Core Points**: Have at least MinPts neighbors within ϵ .
 - **Border Points**: Have fewer than MinPts neighbors but are within ϵ of a core point.
 - **Noise**: Points that are neither core nor border.
2. Form clusters by connecting core points and including border points.

Advantages:

- Detects clusters of arbitrary shapes.
- Handles noise effectively.

Disadvantages:

- Sensitive to parameter selection (ϵ and MinPts).
 - Struggles with varying density clusters.
-

Cluster Evaluation

Overview

Cluster evaluation assesses the quality of clusters generated by an algorithm. This is particularly challenging in unsupervised learning due to the lack of ground truth labels.

Unsupervised Cluster Evaluation

1. **Cohesion and Separation:**

Unsupervised Cluster Evaluation

1. Cohesion and Separation:

- **Cohesion:** Measures the compactness of clusters. Points within a cluster should be close to each other.

$$Cohesion = \sum_{C_i} \sum_{x,y \in C_i} Proximity(x,y)$$

- **Separation:** Measures the distance between clusters. Points in different clusters should be far apart.

$$Separation = \sum_{C_i, C_j} Proximity(C_i, C_j)$$

2. Proximity Matrix:

- A matrix where each entry represents the distance between two points or clusters.
- Used in hierarchical clustering to decide which clusters to merge.

3. Silhouette Score:

- Combines cohesion and separation into a single metric:

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

○

○ **where:**

- a: Mean distance between a point and other points in the same cluster.
- b: Mean distance between a point and points in the nearest cluster.

○ Values range from -1 to 1, where higher values indicate better clustering.

Scalable Clustering Algorithms

Challenges with Scalability:

1. Computational cost for large datasets.
2. Memory limitations when processing all points simultaneously.

Scalable Solutions:

1. **Mini-Batch K-Means:**
 - Processes small random subsets (mini-batches) of the data instead of the entire dataset at each iteration.
2. **Clustering with MapReduce:**
 - Distributed clustering using frameworks like Hadoop or Spark.
3. **Sampling-Based Techniques:**
 - Perform clustering on a sample and generalize results to the entire dataset.
4. **Grid-Based Clustering:**
 - Divides the data space into a grid and performs clustering within the grid cells.

- Example: CLIQUE (CLustering In QUES).

Web Data Mining(Unit-5)

Introduction

Web data mining refers to the process of extracting useful information and patterns from web data, including content, structure, and usage data. It is a key component of web intelligence, combining data mining techniques with web technologies.

Web Terminology and Characteristics

1. Web Terminology:

- **Web Pages:** Basic building blocks of the web, consisting of HTML and other resources.
- **Hyperlinks:** Links connecting one web page to another, forming the structure of the web.
- **Web Servers:** Hosts that serve web pages to users.
- **Web Crawlers:** Automated programs that traverse the web to collect information.

2. Characteristics of the Web:

- **Dynamic and Heterogeneous:** Contains a mix of structured, semi-structured, and unstructured data.
- **Massive Scale:** Billions of web pages and users generate an enormous volume of data.
- **Interconnected:** Hyperlinks create a complex web structure.
- **Evolving:** Content changes frequently, making it challenging to keep up-to-date.

Types of Web Data Mining

1. Web Content Mining:

- **Definition:** Extracting useful information from the content of web pages, such as text, images, videos, or metadata.
- **Techniques:**
 - Text mining (e.g., natural language processing for summarization or sentiment analysis).
 - Multimedia mining (e.g., image or video analysis).
- **Applications:**
 - Product reviews analysis.
 - Extracting structured data (e.g., contact information).

2. Web Usage Mining:

- **Definition:** Analyzing user interactions with websites to discover usage patterns.
- **Data Sources:**
 - Web server logs (e.g., IP addresses, page requests).
 - Browser cookies.
 - Clickstream data.
- **Applications:**
 - Personalization and recommendation systems.

- Improving website navigation.

3. Web Structure Mining:

- **Definition:** Exploiting the hyperlink structure of the web to find relationships between web pages.
 - **Techniques:**
 - Graph-based methods for analyzing link structures.
 - PageRank and HITS algorithms.
 - **Applications:**
 - Web page ranking.
 - Identifying communities or clusters of related pages.
-

Search Engines

Characteristics

- Search engines are specialized systems designed to locate and rank relevant information on the web.
- **Key Features:**
 1. **Scalability:** Must handle billions of web pages and queries.
 2. **Relevance:** Provide results tailored to user intent.
 3. **Speed:** Return results in milliseconds.

Functionality

1. **Crawling:**
 - Automated programs (crawlers or spiders) collect web pages by following hyperlinks.
 2. **Indexing:**
 - Extract and store content in a structured format for fast retrieval.
 - Common indexing methods: Inverted index, forward index.
 3. **Query Processing:**
 - Understand user queries and match them with relevant content in the index.
 4. **Ranking:**
 - Assign a score to web pages based on relevance to the query and other factors.
-

Architecture of Search Engines

1. **Web Crawler:**
 - Traverses the web to fetch pages and store their content.
 2. **Indexer:**
 - Analyzes page content and builds an index for fast retrieval.
 3. **Query Processor:**
 - Parses user queries and matches them with indexed content.
 4. **Ranking Module:**
 - Determines the order of results based on ranking algorithms.
 5. **User Interface:**
 - Presents search results to the user.
-

Ranking of Web Pages

1. PageRank Algorithm:

- Developed by Google, it measures the importance of a web page based on incoming links.
- Formula:

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

Where:

- $PR(A)$: PageRank of page A .
- $L(A)$: Set of pages linking to A .
- $C(i)$: Number of outgoing links from page i .
- d : Damping factor (typically 0.85).

1.

2. HITS Algorithm (Hyperlink-Induced Topic Search):

- Distinguishes between **hubs** (pages with links to many authoritative pages) and **authorities** (pages linked by many hubs).

3. Other Factors:

- Relevance to query.
- Freshness of content.
- User engagement metrics (e.g., click-through rate).

Enterprise Search

Definition

Enterprise search refers to the process of retrieving information from an organization's internal databases, documents, and other resources.

Characteristics:

1. Handles structured and unstructured data.
2. Often restricted to employees with specific access permissions.
3. Focused on specific organizational needs (e.g., technical documentation, financial records).

Applications:

- Enhancing productivity by making information retrieval more efficient.
- Knowledge management systems for large organizations.