DSBDA UNIT – 4 PYQ'S

➢ *MAY / JUN 2022*

Q3)

a) Explain various data pre-processing steps. Discuss essential python libraries for preprocessing. [8]

Data Pre-processing Steps:

1. **Removing Duplicates**:
   Duplicate rows are identified and removed using drop_duplicates() to ensure data integrity.
2. **Handling Missing Data**:
   Missing values are handled using dropna(), fillna() or imputation techniques like mean/median replacement.
3. **Data Transformation**:
   Data is transformed using functions (apply()) or mappings (map(), replace()) to convert values into a more usable form.
4. **Encoding Categorical Data**:
   Convert categorical variables into numeric using methods like **Label Encoding** or **One-Hot Encoding** (get_dummies()).
5. **Feature Scaling**:
   Normalize/standardize data using techniques like **Min-Max Scaling** or **Z-score Standardization** for uniformity.
6. **Outlier Detection & Removal**:
   Detect and handle outliers using statistical methods (e.g., IQR or Z-score techniques).

Essential Python Libraries for Preprocessing:

1. **Pandas**:
   Used for data manipulation – handling missing values, transformation, and encoding.
2. **NumPy**:
   Supports numerical operations, array handling, and working with missing data.
3. **Scikit-learn (sklearn.preprocessing)**:
   Offers preprocessing utilities like StandardScaler, LabelEncoder, OneHotEncoder, SimpleImputer.
4. **Matplotlib/Seaborn** (for EDA):
   Help in visualizing data distributions and identifying preprocessing needs.

b) What are association rules? Explain Apriori Algorithm in brief. [9]

Association Rules:

Association rules are **if-then statements** that identify relationships between items in large datasets. They are widely used in **market basket analysis** to find patterns like:

If a customer buys bread, they are likely to buy butter.

Each rule has three key metrics:

- **Support**: Frequency of itemset in the dataset.
- **Confidence**: Likelihood that item Y is bought when item X is bought.
- **Lift**: Strength of the association between items (Confidence / Expected Confidence).

Apriori Algorithm (Brief):

Apriori is a **frequent itemset mining algorithm** based on the principle that:

If an itemset is frequent, all its subsets are also frequent.

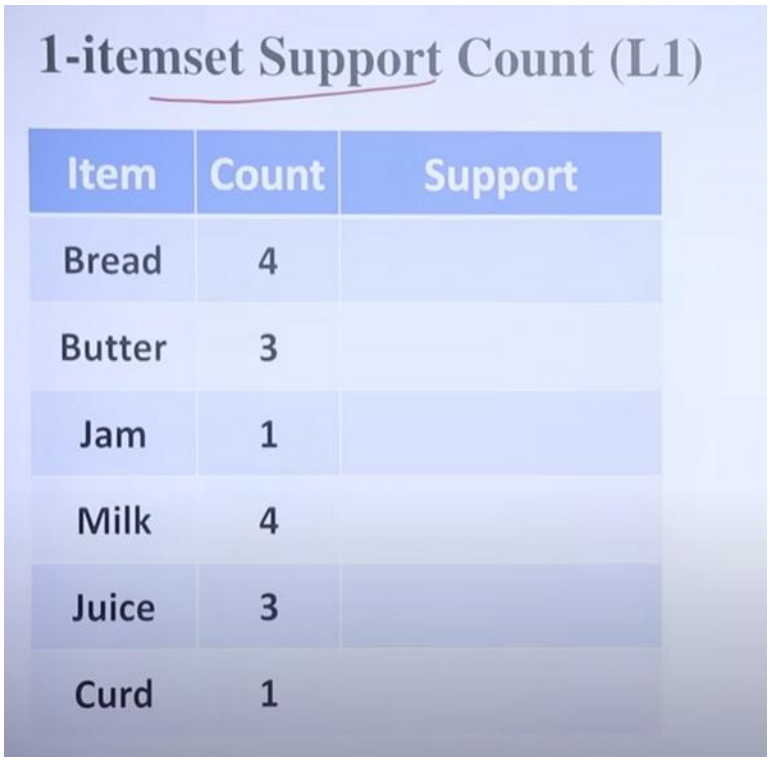
Steps:

1. **Start with 1-itemsets** and count their frequency.
2. **Eliminate** itemsets below **minimum support**.
3. **Generate candidate k-itemsets** from frequent (k−1)-itemsets.
4. **Prune** candidates having infrequent subsets.
5. Repeat until no new frequent itemsets.
6. **Generate association rules** from frequent itemsets using **confidence threshold**.

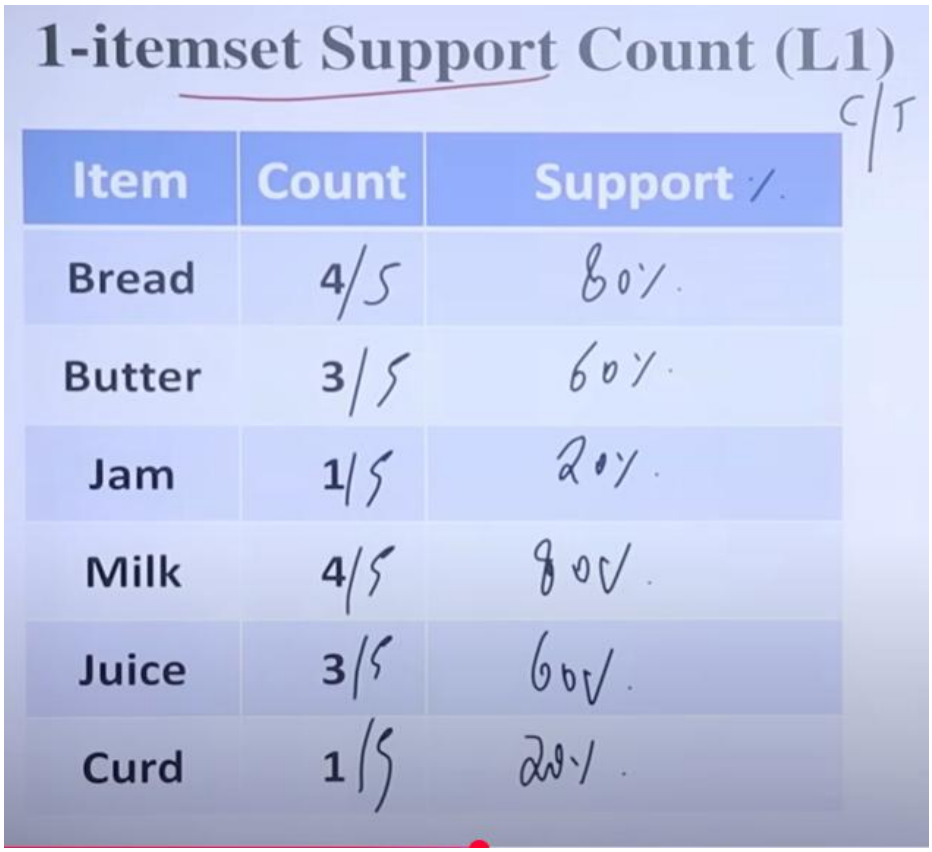
Example:

If {Bread, Butter} is frequent, and Bread is frequent, we can form a rule:

Bread ⇒ Butter with calculated confidence and lift.



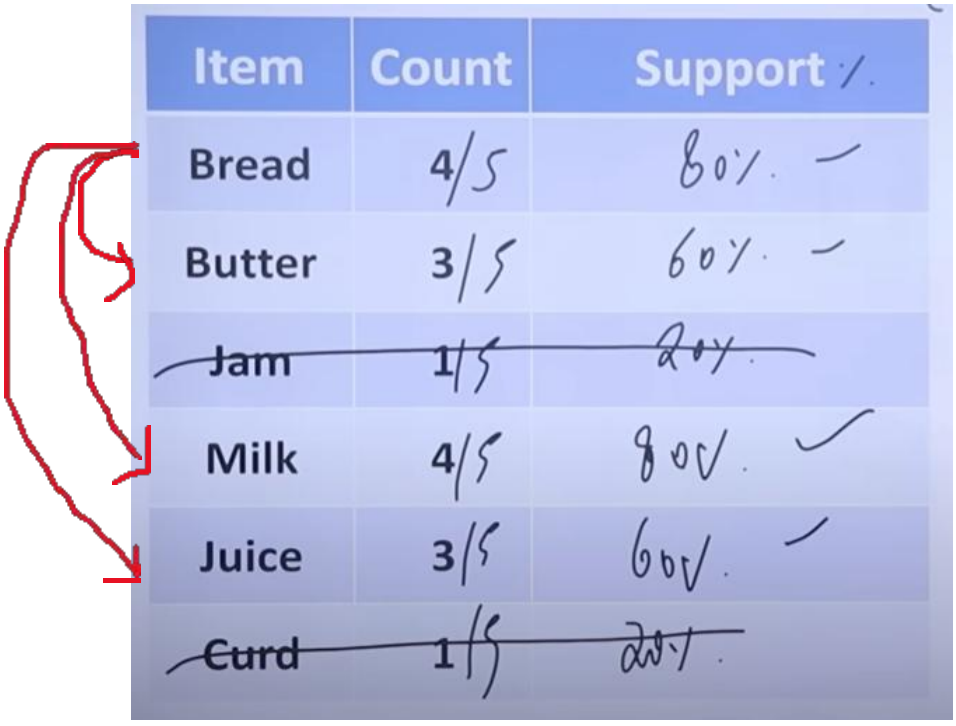
$$\text{Support}(X) = \frac{\text{Number of transactions containing X}}{\text{Total number of transactions}}$$



1-itemset Support Count (L1)

C/T

Item	Count	Support %
Bread	4/5	80%
Butter	3/5	60%
Jam	1/5	20%
Milk	4/5	80%
Juice	3/5	60%
Curd	1/5	20%



Item	Count	Support %
Bread	4/5	80% ✓
Butter	3/5	60% ✓
~~Jam~~	~~1/5~~	~~20%~~
Milk	4/5	80% ✓
Juice	3/5	60% ✓
~~Curd~~	~~1/5~~	~~20%~~

TAKE THE SUPPORT WHERE IT IS MORE THAN 50 %

i.e removed jam and curd because it has support less than 50 %

2-itemsets Support Count (L2)

Item	Count	Support
{Bread, Butter}	2	
{Bread, Milk}	3	
{Bread, Juice}	2	
{Butter, Milk}	3	
{Butter, juice}	1	
{Milk, Juice}	2	

NOW MAKE THE GROUP BREAD → BUTTER
BREAD → MILK
BREAD → JUICE AND SO ON !!
AND COUNT HOW MANY TIME THIS SET OF 2 IS REPEATED

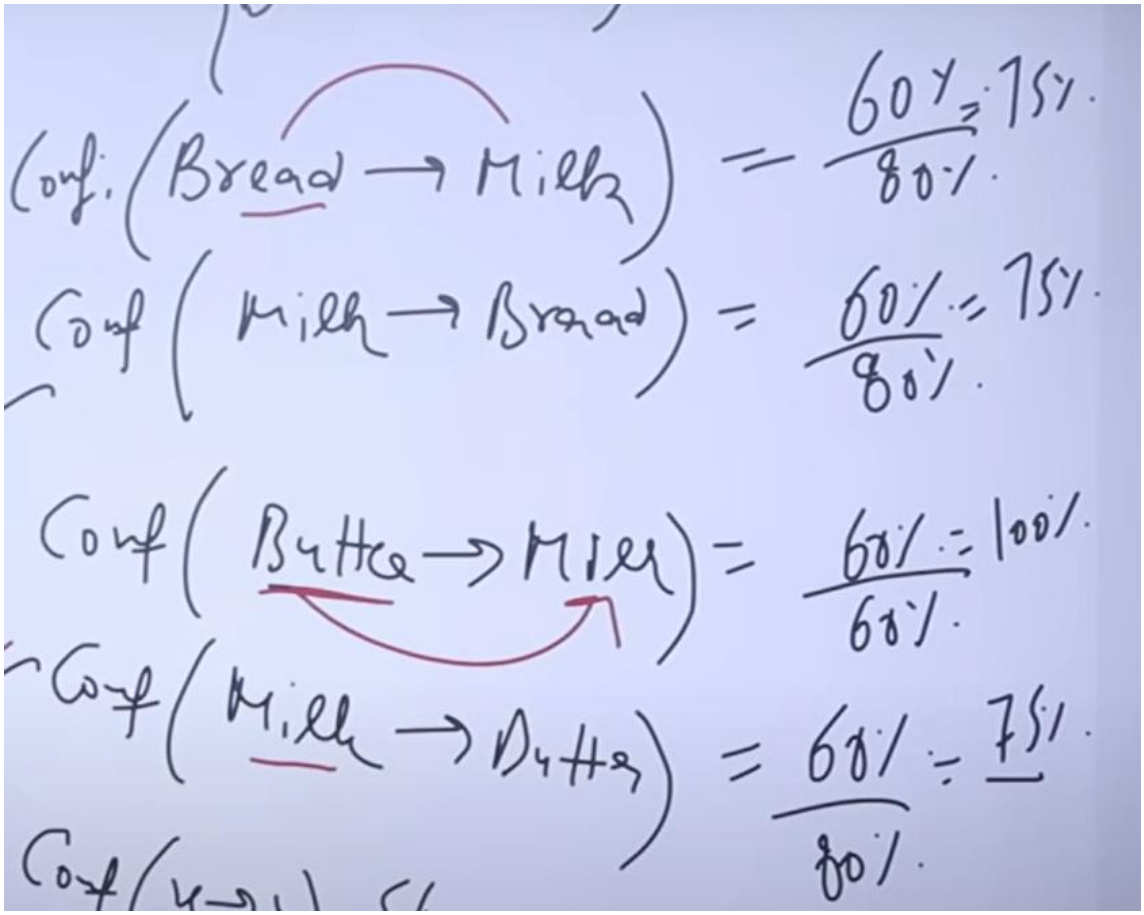
2-itemsets Support Count (L2)

Item	Count	Support %
{Bread, Butter}	2/5	40%
{Bread, Milk}	3/5	60%
{Bread, Juice}	2/5	40%
{Butter, Milk}	3/5	60%
{Butter, juice}	1/5	20%
{Milk, Juice}	2/5	40%

AGAIN SUPPORT IS CALCULATED !!

Item	Count	Support %
~~{Bread, Butter}~~	~~2/5~~	~~40%~~
{Bread, Milk}	3/5	60%
~~{Bread, Juice}~~	~~2/5~~	~~40%~~
{Butter, Milk}	3/5	60%
~~{Butter, juice}~~	~~1/5~~	~~20%~~
~~{Milk, Juice}~~	~~2/5~~	~~40%~~

LESS THAN 50 % SUPPORT REMOVED AGAIN AS DONE PREVIOUSLY !



- **Rule: Bread ⇒ Milk**
  - **Confidence = Support(Bread ∩ Milk) / Support(Bread) = 60 %/80% = 75%**
- **Rule: Milk ⇒ Bread**
  - **Confidence = 60%/80% = 75%**

From {Butter, Milk} → Count = 3

- **Rule: Butter ⇒ Milk**
  - **Confidence = 60 %/60% = 100%**
- **Rule: Milk ⇒ Butter**
  - **Confidence = 60% /80% = 75%**

HENCE IN QUESTION CONFIDENCE WAS ASKED 75 %
HERE ALL CONFIDENCE IS 75 % AND MORE . !

Q4) a) Explain the following:

i) Linear Regression:

Linear Regression is a supervised learning algorithm used for predicting continuous values. It models the relationship between an independent variable X and a dependent variable Y using a straight line:

$$Y = mX + c$$

Where:

- m is the slope (coefficient),
- c is the intercept.

Example: Predicting house price based on area.
Library: sklearn.linear_model.LinearRegression
Goal: Minimize the error (cost function) using techniques like Least Squares.

ii) Logistic Regression:

Logistic Regression is a supervised learning algorithm used for binary classification (output = 0 or 1). It predicts the probability of a class using a sigmoid function:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(mX+c)}}$$

Example: Spam detection, Disease prediction (Yes/No)
Output: Probability between 0 and 1, mapped to classes using a threshold (e.g., 0.5)
Library: sklearn.linear_model.LogisticRegression

Use Case: Spam detection, Disease prediction (Yes/No), Loan approval**.**

Q4)

b) Explain Scikit-learn library for matplotlib with example. [9]

Scikit-learn Overview:

Scikit-learn is a popular **Python machine learning library** that provides simple and efficient tools for **data mining**, **data preprocessing**, **classification**, **regression**, **clustering**, and **model evaluation**. It is built on top of **NumPy**, **SciPy**, and **matplotlib**.

Matplotlib Integration:

While Scikit-learn itself does not do data visualization, it works **in combination with matplotlib** to visualize data, predictions, evaluation results (like confusion matrices), and learning curves.

🔸 Example: Visualizing Linear Regression using matplotlib and scikit-learn

python Copy Edit

```python
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import numpy as np

# Sample data
X = np.array([[1], [2], [3], [4], [5]])
y = np.array([2, 4, 5, 4, 5])

# Model
model = LinearRegression()
model.fit(X, y)

# Prediction
y_pred = model.predict(X)

# Plotting using matplotlib
plt.scatter(X, y, color='blue', label='Actual Data')
plt.plot(X, y_pred, color='red', label='Regression Line')
plt.xlabel("X - Input")
plt.ylabel("y - Output")
plt.title("Linear Regression Example")
plt.legend()
plt.show()
```

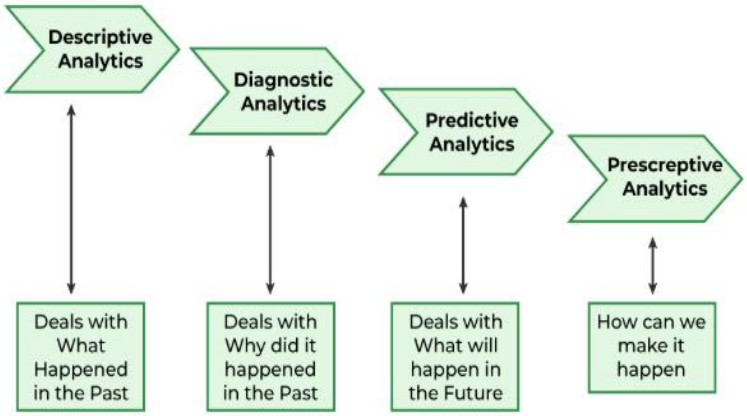
- matplotlib.pyplot is used to **plot graphs** like scatter plots, regression lines, confusion matrices, etc.
- scikit-learn generates predictions and models, which can then be **visualized using matplotlib**.
- This helps in better understanding model performance and behavior.

➢ **NOV / DEC 2022**

Q3

a) What are the types of analytics in big data? Explain in brief. [9]

Types of Analytics in Big Data:
Big Data Analytics is broadly classified into four types. Each type plays a key role in the data analysis process, from understanding the past to predicting the future and recommending actions.



1. Descriptive Analytics

- **Definition**: Descriptive analytics focuses on summarizing historical data to understand what has happened in the past.
- **Purpose**: It helps businesses understand trends, patterns, and behaviors by converting raw data into meaningful information.
- **Techniques**: Reports, dashboards, data aggregation, data visualization.
- **Example**: An e-commerce company analyzing monthly sales figures or customer purchases over the last year.
- **Use Case**: Monitoring website visits, generating financial summaries, and customer feedback reports.

2. Diagnostic Analytics

- **Definition**: Diagnostic analytics goes a step further to investigate the causes of events or outcomes. It explains why something happened.
- **Purpose**: Helps in root cause analysis by identifying correlations, anomalies, and patterns in the data.
- **Techniques**: Drill-down, data mining, correlation analysis.
- **Example**: Analyzing why customer churn increased last month by studying service complaints and competitor offers.
- **Use Case**: Understanding reasons for product failure, drop in revenue, or reduced customer satisfaction.

3. Predictive Analytics

- **Definition**: Predictive analytics uses historical data and machine learning algorithms to forecast future outcomes.
- **Purpose**: To anticipate what is likely to happen in the future based on past data trends.
- **Techniques**: Regression, classification, time series forecasting, machine learning models.
- **Example**: A telecom company predicting which customers are most likely to cancel their subscription.
- **Use Case**: Risk assessment, demand forecasting, fraud detection, and personalized marketing.

4. Prescriptive Analytics

- **Definition**: Prescriptive analytics provides recommendations on actions to take for desired outcomes, using optimization and simulation techniques.
- **Purpose**: To suggest the best decision based on predicted scenarios and business objectives.
- **Techniques**: Decision trees, optimization algorithms, simulations.
- **Example**: A logistics company using prescriptive analytics to choose the most efficient delivery routes.
- **Use Case**: Strategic planning, inventory management, dynamic pricing, and scheduling.

b) Calculate the support and confidence value for all the possible item sets. **[9]**

Transaction ID	Items bought
1	Onion, Potato, Cold drink
2	Onion, Burger, Cold drink
3	Eggs, Onion, Cold drink
4	Potato, Milk, Eggs.
5	Potato, Burger, cold drink, Milk eggs.

Transactions (Total = 5):

TID	Items Bought
1	Onion, Potato, Cold drink
2	Onion, Burger, Cold drink
3	Eggs, Onion, Cold drink
4	Potato, Milk, Eggs
5	Potato, Burger, Cold drink, Milk, Eggs

1. Single Item Support (%)

Item	Count	Support (%)
Onion	3	60%
Potato	3	60%
Cold drink	4	80%
Burger	2	40%
Eggs	3	60%
Milk	2	40%

Step 2: Support for All Pair Itemsets (2-item sets)

Pair Itemset	Count	Support (%)
Onion, Potato	1	20%
Onion, Cold drink	3	60%
Onion, Burger	1	20%
Onion, Eggs	1	20%
Potato, Cold drink	2	40%
Potato, Milk	2	40%

Potato, Burger	1	20%
Potato, Eggs	2	40%
Cold drink, Burger	2	40%
Cold drink, Eggs	2	40%
Cold drink, Milk	1	20%
Burger, Eggs	1	20%
Eggs, Milk	2	40%
Burger, Milk	1	20%

Step 3: Confidence for Selected Rules (X → Y)

Rule	Calculation (Support X ∪ Y / Support X)	Confidence (%)
Onion → Cold drink	60% / 60%	100%
Cold drink → Onion	60% / 80%	75%
Potato → Cold drink	40% / 60%	66.67%
Cold drink → Potato	40% / 80%	50%
Burger → Cold drink	40% / 40%	100%
Cold drink → Burger	40% / 80%	50%
Potato → Milk	40% / 60%	66.67%
Milk → Potato	40% / 40%	100%
Eggs → Milk	40% / 60%	66.67%
Milk → Eggs	40% / 40%	100%
Potato → Eggs	40% / 60%	66.67%
Eggs → Potato	40% / 60%	66.67%
Burger → Milk	20% / 40%	50%
Milk → Burger	20% / 40%	50%

Summary Table:

Itemset / Rule	Support (%)	Confidence (%) (X→Y)
Onion	60%	Onion → Cold drink: 100%
Potato	60%	Potato → Cold drink: 66.67%
Cold drink	80%	Burger → Cold drink: 100%
Burger	40%	Burger → Milk: 50%
Eggs	60%	Eggs → Milk: 66.67%
Milk	40%	Milk → Potato: 100%
Onion, Cold drink	60%	Cold drink → Onion: 75%
Potato, Milk	40%	Potato → Milk: 66.67%
Potato, Eggs	40%	Eggs → Potato: 66.67%
Burger, Milk	20%	Milk → Burger: 50%

Q4)

a) Explain the use of logistic function in logistic regression in detail. [9]

Logistic regression is a statistical method used for binary classification problems — where the outcome can be one of two classes (e.g., yes/no, 0/1, spam/not spam).

The core component of logistic regression is the logistic function (also called the sigmoid function), which is used to model the probability that a given input belongs to a particular class.

- The logistic function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- Here, **z** is a linear combination of input features and their coefficients:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

- The logistic function takes any real number **z** and outputs a value between 0 and 1.

Why use the Logistic Function in Logistic Regression

1. **Probability Output:**
    - The logistic function maps real-valued inputs to the (0,1) interval, which can be interpreted as the **probability** of belonging to the positive class.
2. **Non-linear Transformation:**
    - Unlike linear regression which can predict values beyond [0,1], logistic regression uses the logistic function to squash values into a valid probability range.
3. **Decision Boundary:**
    - By applying a threshold (commonly 0.5), logistic regression classifies the output into class 0 or 1.
4. **Interpretability:**
    - The output probability helps in understanding the confidence level of classification.
5. **Mathematical Convenience:**
    - The logistic function is differentiable and smooth, enabling optimization techniques like gradient descent to find the best coefficients β\betaβ.

b) Write short note on the following:

i) Removing duplicates from data set.

ii) Handling missing data

iii) Data transformation

i) Removing duplicates from data set.

Removing duplicates is an important **data cleaning** step to ensure accuracy and consistency in analysis.

- **Duplicates** occur when the same data row appears more than once in the dataset.
- Keeping duplicates can **bias the results** and affect model training.
- Tools like **Pandas in Python** provide functions like drop_duplicates() to remove them easily.
- It's essential to decide whether to drop full-row duplicates or check based on specific columns.

---

ii) Handling Missing Data

Missing data can lead to incorrect insights or model errors, so handling it properly is critical. Common strategies include:

- **Deletion**: Remove rows or columns with too many missing values.
- **Imputation**:
    - Use mean/median/mode to fill numerical data.
    - Use most frequent or constant values for categorical data.
- **Advanced methods**: Use KNN, regression, or ML models to predict missing values.
  Tools like Pandas offer fillna() and dropna() for handling missing data.

---

iii) Data Transformation

Data transformation refers to **modifying data to improve its quality or fit for modeling**.
Common transformations include:

- **Normalization/Standardization**: Scale features to a specific range or mean-zero.
- **Encoding**: Convert categorical data into numeric format (e.g., one-hot encoding).
- **Log/Power Transform**: Reduce skewness in numeric data.
- **Aggregation**: Summarize data (e.g., total sales per month).
  Transformation helps in improving model performance and interpretability.

➢ **MAY / JUN 2023**

Q3)

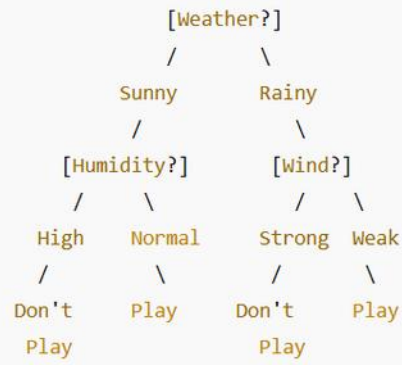
a) Explain why decision tree are used. Draw a sample decision tree and explain its parts.[9]

Decision trees are popular in machine learning for the following reasons:

1. Easy to Understand & Interpret:
    - Decision trees mimic human decision-making with a clear flowchart-like structure.
2. Handles Both Numerical and Categorical Data:
    - Suitable for diverse datasets.
3. No Need for Data Scaling:
    - No normalization or standardization needed.
4. Feature Selection Built-in:
    - It automatically selects the most significant features at each node.
5. Used for Both Classification and Regression:
    - Decision trees work for predicting categories (classification) and numbers **(regression).**

✅ Sample Decision Tree Diagram:

mathematica Copy Edit



Explanation of Parts:

1. **Root Node**:
    - The top node where the decision process starts.
    - *Example: "Weather?"*
2. **Internal Nodes**:
    - Represent questions or decisions based on features.
    - *Example: "Humidity?", "Wind?"*
3. **Branches/Edges**:
    - Show the outcome of a decision or test.
    - *Example: "Sunny", "Rainy", "High", "Normal"*
4. **Leaf Nodes (Terminal Nodes)**:
    - Represent final output or class label.
    - *Example: "Play", "Don't Play"*

b) How Apriori Algorithm works, explain with suitable example? [9]

ALREADY DONE , WRITE ALGORITH AND EXPLAIN THE EXAMPLE PREVIOUSLY DONE (i.e milk bread wala) !!

Q4

a) What is Data Preprocessing? Explain in detail about handling missing data and transformation of data. [9]

Data preprocessing is the process of cleaning and transforming raw data into a useful and understandable format before feeding it to a machine learning model.
It ensures the data is accurate, complete, and suitable for analysis.

Handling Missing Data:

Missing data can negatively affect the performance of models. There are several techniques to handle it:

1. **Removing Data**
    - Remove rows or columns that contain missing values.
    - Use only if the missing percentage is low.
2. **Imputation Techniques**
    - **Mean/Median/Mode Imputation**: Replace missing values with the column's mean, median, or mode.
    - **Forward/Backward Fill**: Propagate previous or next value in time-series data.
    - **KNN or Regression Imputation**: Use machine learning models to predict missing values.
3. **Flagging**
    - Add a new column to indicate if data was missing — helps retain information.

Data Transformation:

Transformation converts data into a suitable format or structure for analysis:

1. **Normalization (Min-Max Scaling)**
    - Rescales data between 0 and 1.
    - Useful when features have different ranges.
2. **Standardization (Z-score Scaling)**
    - Rescales data with mean = 0 and standard deviation = 1.
3. **Encoding Categorical Variables**
    - **Label Encoding**: Assigns a numeric value to each category.

- **One-Hot Encoding**: Creates binary columns for each category.

4. **Log/Power Transformation**
    - Applied to skewed data to make it more normal.

Q4

b) Explain Naïve Bayes' Classifier and Its Applications [9]

Naïve Bayes is a supervised learning algorithm based on Bayes' Theorem with a strong assumption of feature independence.

It is used for classification tasks and works well with large datasets and text data.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- **P(A|B)**: Posterior Probability (probability of class A given features B)
- **P(B|A)**: Likelihood (probability of features B given class A)
- **P(A)**: Prior probability of class A
- **P(B)**: Prior probability of features B

How Naïve Bayes Works:

1. Calculate prior probability of each class from training data.
2. Calculate the likelihood of input features for each class.
3. Apply Bayes' Theorem to compute the posterior for each class.
4. Assign the class with the highest posterior probability.

Types of Naïve Bayes:

- Gaussian Naïve Bayes: For continuous data (assumes normal distribution)
- Multinomial Naïve Bayes: For text classification, word counts
- Bernoulli Naïve Bayes: For binary/boolean features

Applications of Naïve Bayes:

1. **Spam Filtering**: Classifies emails as spam or not spam.
2. **Sentiment Analysis**: Determines positive or negative sentiment in reviews.
3. **Document Categorization**: Classifies documents or articles into topics.
4. **Medical Diagnosis:** Predicts diseases based on symptoms.
5. **Fraud Detection**: Identifies suspicious transactions.

- **NOV / DEC 2023**

All question are repeated !!!

- **MAY / JUN 2024**

Q3

a) What is Logistic Regression, and how does it differ from Linear Regression? What is the Sigmoid Function, and what role does it play in Logistic Regression? [9]

Logistic Regression is a **supervised classification algorithm** used to predict the probability of a binary outcome (yes/no, 0/1, true/false).

It predicts the probability that a given input point belongs to a particular category or class.

Difference between Logistic and Linear Regression:

Feature	Linear Regression	Logistic Regression
Output	Continuous value	Probability (0 to 1) / Binary outcome
Function Used	Linear Equation (Y = mx + c)	Sigmoid Function (S-shaped curve)
Problem Type	Regression problems	Classification problems
Output Range	(-∞ to +∞)	(0 to 1)

Sigmoid Function

The **sigmoid function** is a mathematical function used to map predicted values into a probability range of **0 to 1**:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Where z is the linear combination of input features.

Role of Sigmoid in Logistic Regression:

- Converts linear output (z) into a probability.
- If the output is **> 0.5**, class is predicted as **1** (positive class).
- If the output is **≤ 0.5**, class is predicted as **0** (negative class).
- Helps in binary classification using threshold-based decisions.

Example:

If logistic regression predicts σ(z)=0.8\sigma(z) = 0.8σ(z)=0.8, this means there's an 80% chance the input belongs to class 1.

Q3 b)
do it by yourself !!!

Q4

a) How does the Apriori algorithm discover frequent itemsets in a dataset? What is the role of support and confidence in the context of association rule mining using the Apriori algorithm?

Apriori Algorithm Overview:

The Apriori algorithm is a fundamental algorithm used for **association rule mining** in data mining. It discovers **frequent itemsets** (sets of items that appear together often in transactions) and derives **association rules** to find interesting relationships in data.

Steps of the Apriori Algorithm:

1. **Generate Candidate Itemsets:**
    - Start with all 1-itemsets (individual items).

- Iteratively generate k-itemsets (itemsets of size k) from (k–1)-itemsets.

2. **Prune Infrequent Itemsets:**
   - Remove itemsets whose **support** is less than the minimum support threshold.
   - Uses **Apriori Property**: If an itemset is frequent, all its subsets must also be frequent.
3. **Repeat Until No New Frequent Itemsets:**
   - Keep generating and pruning itemsets until no further frequent itemsets are found.
4. **Generate Association Rules:**
   - From frequent itemsets, generate rules like A → B where A and B are subsets.
   - Use **confidence** to measure rule strength.

Role of Support:

- **Support** measures how frequently an itemset appears in the dataset.
- Formula:

$$\text{Support}(A) = \frac{\text{Number of transactions containing A}}{\text{Total number of transactions}}$$

Role of Confidence:

- **Confidence** indicates how often items in B appear in transactions that contain A.
- Formula:

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)}$$

Apriori uses support to find frequent itemsets and confidence to generate strong rules. It is widely used in market basket analysis to uncover buying patterns and product associations.

Q4 b) Explain the process of building a decision tree? What are the criteria used for splitting nodes in a decision tree? [9]

1. Building a Decision Tree (Process):

A decision tree is built using a **top-down, recursive partitioning** approach:

1. **Start with the Entire Dataset:**
    - Begin with all training data at the root node.
2. **Select the Best Attribute:**
    - Use a **splitting criterion** (like Information Gain, Gini Index) to choose the attribute that best separates the data.
3. **Split the Data:**
    - Partition the data based on the selected attribute's values.
4. **Repeat Recursively:**
    - Repeat the process for each child node with the remaining attributes and data until:
        - All instances belong to the same class (pure node), or
        - There are no more attributes left, or
        - A stopping condition (like minimum number of samples) is met.
5. **Assign Leaf Nodes:**
    - Leaf nodes are labeled with the majority class of the data in that node.

2. Criteria for Splitting Nodes:

Several criteria are used to determine the "best" attribute for splitting:

Criterion	Description
Information Gain	Measures the reduction in entropy. Higher gain = better split. Used in ID3.
Gain Ratio	Improves Information Gain by penalizing attributes with many values. Used in C4.5.
Gini Index	Measures impurity in data. Lower Gini = better split. Used in CART.
Chi-square	Statistical test for independence between attribute and target.
Reduction in Variance	Used in regression trees. Chooses splits that reduce output variance the most.

➢ **NOV / DEC 2024**

Q3) a) Define and explain Entropy and Information gain. Calculate the entropy of the following distribution **[9]**

Fruit Color	Taste	Count
Yellow	Sweet	10
Red	Sweet	5
Green	sour	15
Orange	sour	5

Entropy and Information Gain

Entropy:

Entropy is a measure of the uncertainty or impurity in a dataset. It helps in deciding which attribute to split on in decision trees.

The formula for entropy:

$$\text{Entropy}(S) = -\sum p_i \log_2(p_i)$$

Where:

- pi is the probability of class iii
- Base 2 log is used.

Information Gain:

It measures the reduction in entropy after a dataset is split on an attribute.

$$\text{Information Gain} = \text{Entropy}(S) - \sum \left(\frac{|S_i|}{|S|} \cdot \text{Entropy}(S_i) \right)$$

Given Table:

Fruit Color	**Taste**	**Count**
Yellow	Sweet	10

Red	Sweet	5
Green	Sour	15
Orange	Sour	5

Step 1: Total Count = 10 + 5 + 15 + 5 = 35

Sweet: 10 + 5 = 15

Sour: 15 + 5 = 20

Step 2: Calculate probabilities

$$P(\text{Sweet}) = 15/35 = 0.4286$$

$$P(\text{Sour}) = 20/35 = 0.5714$$

Step 3: Entropy Calculation

$$\begin{aligned}\text{Entropy} &= -(0.4286 \cdot \log_2(0.4286) + 0.5714 \cdot \log_2(0.5714)) \\ &= -(0.4286 \cdot -1.2224 + 0.5714 \cdot -0.8074) \\ &= -(-0.5239 - 0.4615) = 0.9854\end{aligned}$$

Final Answer:

- **Entropy of the distribution = 0.9854 bits**
- This value shows moderate impurity in the dataset.

b) Explain naive bayes classifire [8]

Naïve Bayes is a **probabilistic classification algorithm** based on **Bayes' Theorem** with a **naïve assumption** that all features are independent of each other 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 features **X**
- $P(X|C)$ = Likelihood of features **X** given class **C**
- $P(C)$ = Prior probability of class **C**
- $P(X)$ = Evidence (overall probability of X)

Working Steps:

1. **Calculate prior probability** for each class from training data.
2. **Calculate likelihood** for each feature given the class.
3. **Multiply the probabilities** (assuming feature independence).
4. **Choose the class** with the highest posterior probability.

Types of Naïve Bayes:

- **Gaussian Naïve Bayes** – for continuous data (assumes Gaussian distribution).
- **Multinomial Naïve Bayes** – for discrete counts (e.g., word frequency).
- **Bernoulli Naïve Bayes** – for binary/boolean features.

Applications:

- Spam detection
- Text classification
- Sentiment analysis
- Medical diagnosis

Advantages:

- Simple and fast
- Works well with high-dimensional data
- Requires less training data

Q4

a) , b)

Both are repeated !!!