AIM:- Study of preprocessing methods.

Write a program to find following statistics from a given dataset. Mean, mode, median, variance, standard deviation, quartiles, interquartile range.

Data Preprocessing in Machine Learning

1. Introduction

Data preprocessing is the technique of converting raw, messy, and unstructured data into a **clean, consistent, and usable dataset**. In machine learning, real-world data is rarely perfect. It is often **incomplete, noisy, inconsistent, and may contain outliers or missing values**. If such data is directly used for training models, it can lead to poor accuracy and unreliable predictions.

Data preprocessing resolves these issues by applying systematic methods to **clean, transform, and scale the data**, ensuring that machine learning models can learn meaningful patterns effectively. It is one of the most important stages in the machine learning pipeline.

2. Key Preprocessing Steps

Handling Missing Values (Data Cleaning)

Real-world datasets often contain missing or null values due to human error, system failure, or incomplete data collection. These missing values must be treated properly to avoid biased or incorrect results.

Common approaches include:

- **Removing records** with missing values (if very few).
- **Imputation**, where missing values are replaced using statistical methods such as **mean, median, or mode**.

This step improves data completeness and reliability.

• Categorical Encoding (Data Transformation)

Machine learning algorithms work with numerical values and cannot directly process textual or categorical data. Therefore, categorical variables must be converted into numeric form.

Label Encoding

Each category is assigned a unique numeric label.
Example: Low = 0, Medium = 1, High = 2
It is mainly used for **ordinal data**, where a natural order exists.

One-Hot Encoding (Dummy Variables)

A new binary column is created for each category.
Example: Color = Red, Blue, Green → Color_Red, Color_Blue, Color_Green
This method is used for **nominal data** (no order) and prevents the model from assuming a false ranking between categories.

• Feature Scaling

Different features often have different units and value ranges. For example, age may range from 0–100, while salary may range from thousands to lakhs. Such differences can cause models to become biased toward larger values. Feature scaling solves this problem.

Normalization

Rescales values into a fixed range, usually **0 to 1**.
It is useful for distance-based algorithms such as KNN and neural networks.

Standardization

Transforms data so that it has a **mean of 0 and a standard deviation of 1**.
It centers the data and is widely used in regression, SVM, and PCA.

• Outlier Detection using IQR Method

Outliers are extreme values that significantly differ from other data points and can distort model performance. The **Interquartile Range (IQR)** method is commonly used to detect them.

$$IQR = Q3 - Q1$$

Lower bound = Q1 − 1.5 × IQR
Upper bound = Q3 + 1.5 × IQR

Values outside this range are treated as outliers and can be removed or capped to improve data quality.

• Missing Value Treatment using Mean, Median, and Mode

This is the most widely used statistical approach for missing data handling:

- **Mean** – Used for numerical data with normal distribution.
- **Median** – Used for skewed data or data with outliers.
- **Mode** – Used for categorical data.

This ensures that the dataset remains complete without losing important records.

Code

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, MinMaxScaler, StandardScaler
data = {
    "Age": [22, 25, np.nan, 28, 35, 30, 24],
    "Salary": [22000, 25000, 27000, np.nan, 120000, 30000, 24000],
    "Gender": ["Male", "Female", "Female", "Male", "Male", np.nan, "Female"],
    "City": ["Ahmedabad", "Surat", "Vadodara", "Ahmedabad", "Surat", "Vadodara", "Surat"]
}
df = pd.DataFrame(data)
print("Original Dataset:\n", df)
df["Age"].fillna(df["Age"].mean(), inplace=True)
df["Salary"].fillna(df["Salary"].median(), inplace=True)
df["Gender"].fillna(df["Gender"].mode()[0], inplace=True)
print("\nAfter Handling Missing Values:\n", df)
le = LabelEncoder()
df["Gender_Label"] = le.fit_transform(df["Gender"])
print("\nAfter Label Encoding:\n", df)
df = pd.get_dummies(df, columns=["City"])
print("\nAfter One-Hot Encoding:\n", df)
scaler = MinMaxScaler()
df[["Age", "Salary"]] = scaler.fit_transform(df[["Age", "Salary"]])

print("\nAfter Normalization:\n", df)
std = StandardScaler()
df[["Age", "Salary"]] = std.fit_transform(df[["Age", "Salary"]])
print("\nAfter Standardization:\n", df)
Q1 = df["Salary"].quantile(0.25)
Q3 = df["Salary"].quantile(0.75)
IQR = Q3 - Q1
```

lower = Q1 - 1.5 * IQR

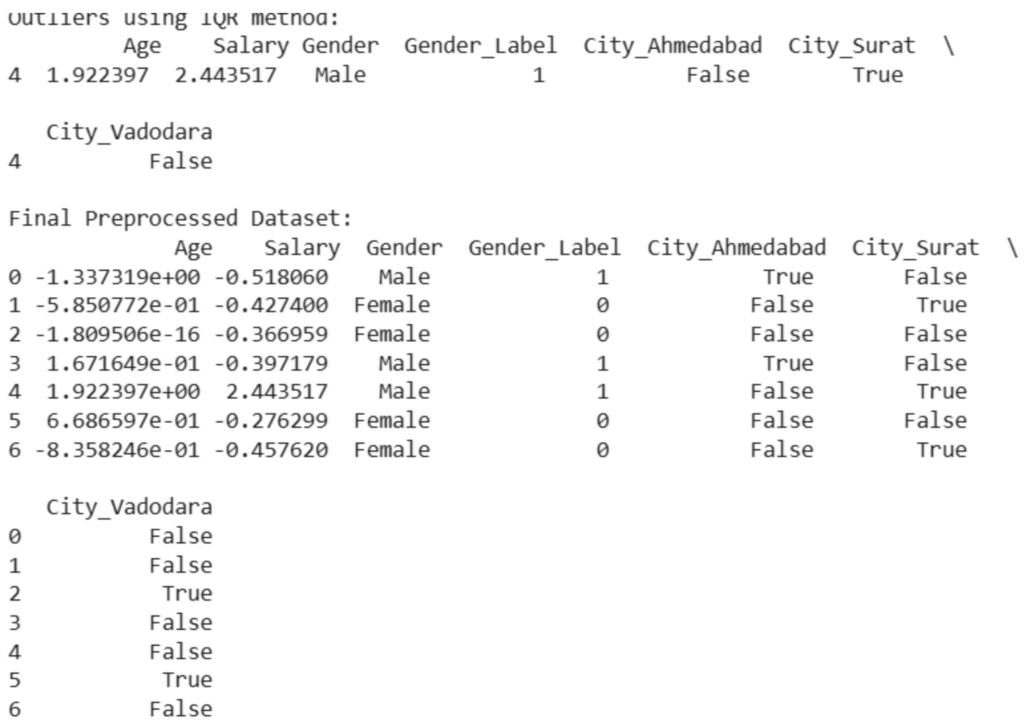
upper = Q3 + 1.5 * IQR

outliers = df[(df["Salary"] < lower) | (df["Salary"] > upper)]

print("\nOutliers detected using IQR:\n", outliers)

print("\nFinal Preprocessed Dataset:\n", df)

output:-



```
Outliers using IQR method:
        Age    Salary Gender  Gender_Label  City_Ahmedabad  City_Surat  \
4  1.922397  2.443517   Male             1           False        True

   City_Vadodara
4          False

Final Preprocessed Dataset:
            Age    Salary  Gender  Gender_Label  City_Ahmedabad  City_Surat  \
0 -1.337319e+00 -0.518060    Male             1            True       False
1 -5.850772e-01 -0.427400  Female             0           False        True
2 -1.809506e-16 -0.366959  Female             0           False       False
3  1.671649e-01 -0.397179    Male             1            True       False
4  1.922397e+00  2.443517    Male             1           False        True
5  6.686597e-01 -0.276299  Female             0           False       False
6 -8.358246e-01 -0.457620  Female             0           False        True

   City_Vadodara
0          False
1          False
2           True
3          False
4          False
5           True
6          False
```