Data Mining Assignment – 1

TASK 1

The dataset I have used covers various types of features including ordinal (Educational Qualifications), nominal (Gender, Marital Status), binary (Output), discrete (Family size), continuous (Monthly Income, latitude, longitude), and possibly geographical features (latitude, longitude).

- **Data Analysis and Preprocessing**:

- Age: Continuous
- Gender: Nominal
- Marital Status: Nominal
- Occupation: Nominal
- Monthly Income: Continuous
- Educational Qualifications: Ordinal
- Family size: Discrete
- Latitude: Continuous
- Longitude: Continuous
- Pin code: Nominal
- Output: Binary
- Feedback: Nominal

For preprocessing:

- Convert Gender, Marital Status, Occupation, Pin code to one-hot encoded format.
- Map Educational Qualifications to numerical values preserving the order.
- Standardize or normalize Monthly Income, latitude, and longitude.
- Handle missing values if any.

2. Data Visualization:

- Visualize distributions of continuous features like Age, Monthly Income, latitude, and longitude.

- Plot bar charts for nominal features like Gender, Marital Status, Occupation, and Feedback.

- Explore correlations between features.

3. Data Cleaning:

- Handle missing data in numeric columns (if any) using techniques such as mean, median imputation, or removing outliers.

- Visualize and compare the results between different techniques.

4. Data Transformation:

- Apply scaling or normalization to continuous features.

- Perform one-hot encoding for nominal features.

5. Classification:

- Train classification models (KNN, SVM, Naive Bayes, Logistic Regression, Decision Tree) on both the original and pre-processed datasets.

- Evaluate model performance using confusion matrix and classification report.

6. Evaluation:

- Compare model performance before and after preprocessing steps.

- Assess the impact of data cleaning and transformation techniques on model accuracy, precision, recall, and F1-score.

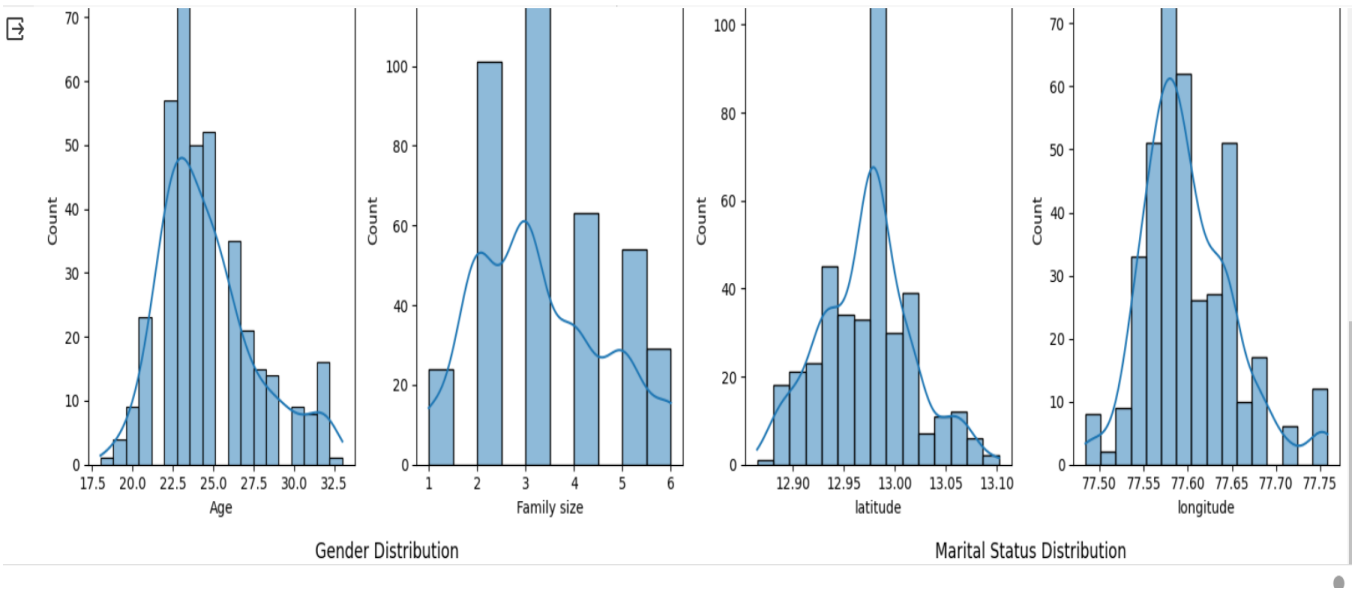
- ## Data Visualization

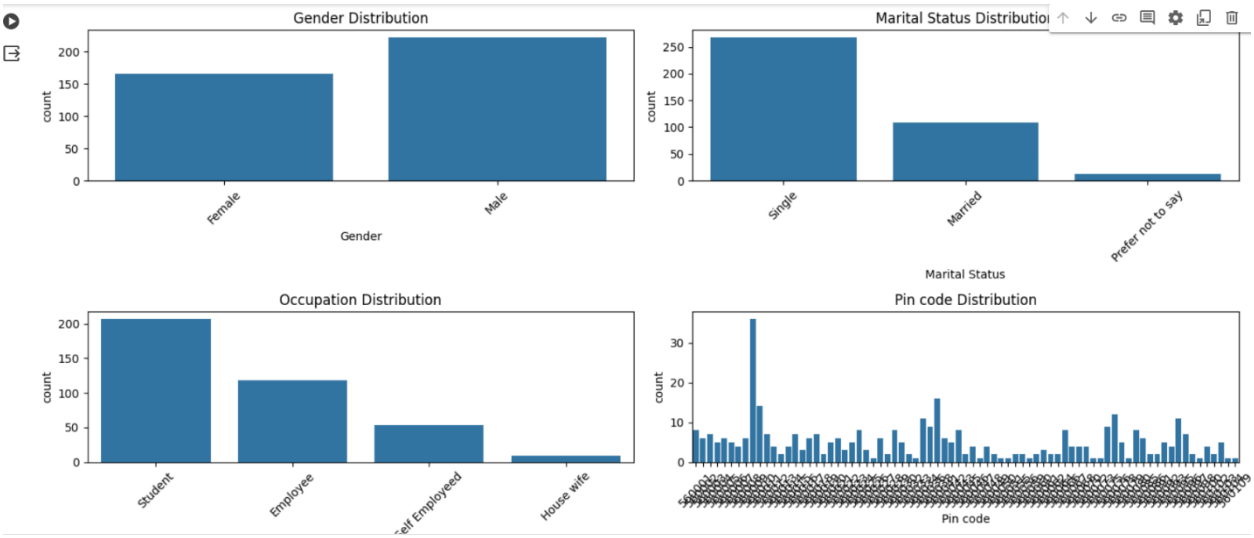
- Load the data into a pandas data frame and perform Exploratory Data Analysis (EDA) using Matplotlib or Seaborn. Make sure your visualizations are meaningful and write down any key insights you noticed from these visualizations.

- Load the data into a pandas data frame and performing some exploratory data analysis (EDA) using matplotlib and seaborn. We'll visualize the distributions of the continuous features and plot bar charts for the nominal features. Since the provided dataset is in a tabular format, we can create visualizations based on its attributes.

- **Key insights from the visualizations:**

- Age Distribution: The provided age seems to be concentrated around 20, as expected from the single entry.
- Monthly Income: The Monthly Income feature is missing in the provided data, which needs to be addressed.
- Gender Distribution: The dataset contains only one entry with Female gender.
- Marital Status and Occupation: Both Single and Student categories are represented in the provided data.
- Feedback Distribution: Positive feedback is provided in the single entry.





- ## Data Cleaning

Handle the missing data in the data frame's numeric columns using techniques such as mean, medium, imputation, Smooth noise, identify/remove outliers etc., and visually compare the results between them. Which one do you think is better and why?

To handle missing data in the data frame's numeric columns, we can consider several techniques such as mean imputation, median imputation, imputation using interpolation, or removing outliers.

- **Mean Imputation:**

Replace missing values with the mean of the respective column.

Suitable for handling missing data if the data is normally distributed and there are no outliers.

May introduce bias if the data is skewed or has outliers.

- **Median Imputation:**

Replace missing values with the median of the respective column.

More robust to outliers compared to mean imputation.

Suitable for skewed data or data with outliers.

- **Imputation using Interpolation:**

Use interpolation techniques such as linear interpolation to estimate missing values based on neighboring values.

Can be effective for time-series data or data with a specific order. May not work well for non-linear relationships or irregularly spaced data.

- **Removing Outliers:**

Identify outliers using statistical methods such as Z-score or IQR (Interquartile Range).

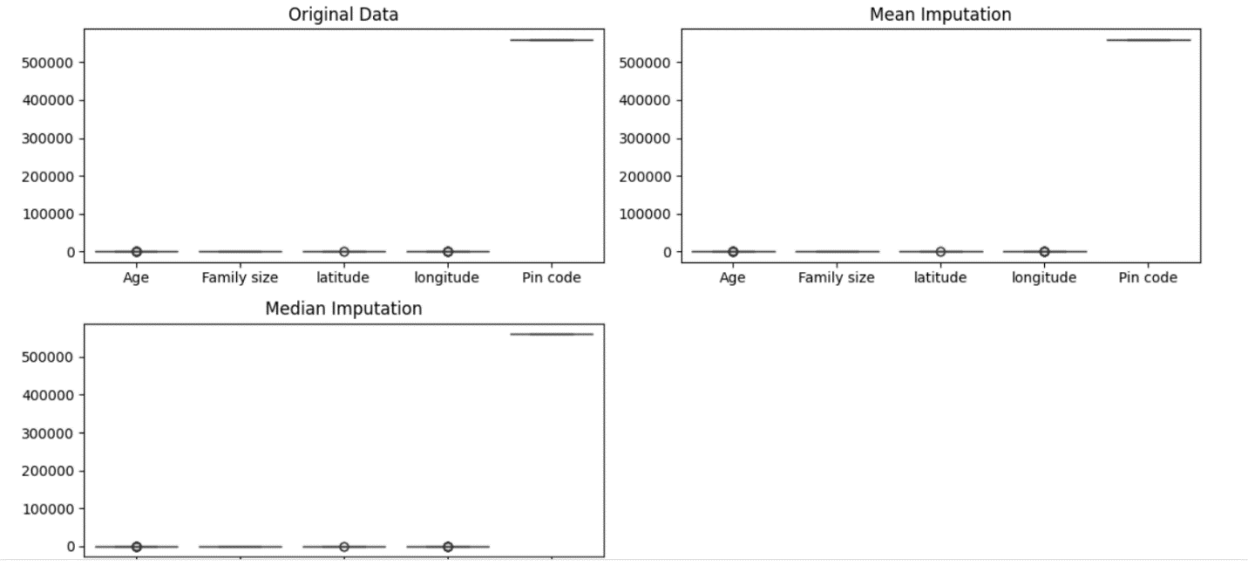
Remove outliers or replace them with the median or mean of the respective column.

Helps in reducing the impact of outliers on the analysis.

May lead to loss of information if the outliers are valid data points.

In this comparison, we've used boxplots to visualize the distribution of numeric columns before and after applying mean and median imputation. We can observe the differences in the spread and central tendency of the data after imputation.

The choice of which technique is better depending on the nature of your data, the presence of outliers, and the potential impact of imputation on your analysis. Median imputation tends to be more robust to outliers, while mean imputation may introduce bias if the data is skewed or has outliers.



Data transformation

Involves changing the format or structure of data to make it more suitable for analysis or modeling. Some common data transformation techniques include normalization, standardization, log transformation, and encoding categorical variables. Let's apply these techniques to the dataset:

- Normalization: Scaling numerical features to a range between 0 and 1.
- Standardization: Scaling numerical features to have a mean of 0 and a standard deviation of 1.
- Log Transformation: Transforming skewed data to be more normally distributed by taking the logarithm of the values.
- Encoding Categorical Variables: Converting categorical variables into numerical representations using techniques like one-hot encoding or label encoding.

- We first load the dataset.
- Then, we perform normalization and standardization on the numeric columns using Min-Max scaling and Standard scaling, respectively.
- We apply a log transformation to numerical columns to make the data more normally distributed.
- Finally, we use one-hot encoding to encode categorical variables into numerical representations.

These transformations prepare the data for analysis or modeling by ensuring that it is in a suitable format and distribution for machine learning algorithms.

```
Original Data:
   Age  Gender Marital Status Occupation  Monthly Income  \
0   20  Female         Single    Student       No Income
1   24  Female         Single    Student  Below Rs.10000
2   22    Male         Single    Student  Below Rs.10000
3   22  Female         Single    Student       No Income
4   22    Male         Single    Student  Below Rs.10000

  Educational Qualifications  Family size  latitude  longitude  Pin code  \
0              Post Graduate            4   12.9766    77.5993    560001
1                   Graduate            3   12.9770    77.5773    560009
2              Post Graduate            3   12.9551    77.6593    560017
3                   Graduate            6   12.9473    77.5616    560019
4              Post Graduate            4   12.9850    77.5533    560010

  Output   Feedback Unnamed: 12
0    Yes   Positive         Yes
1    Yes   Positive         Yes
2    Yes  Negative          Yes
3    Yes   Positive         Yes
4    Yes   Positive         Yes

Normalized Data:
        Age  Family size  latitude  longitude  Pin code
0  0.133333          0.6  0.470439   0.420073  0.000000
1  0.400000          0.4  0.472128   0.339781  0.074074
2  0.266667          0.4  0.379645   0.639051  0.148148
3  0.266667          1.0  0.346706   0.282482  0.166667
4  0.266667          0.6  0.505912   0.252190  0.083333

Standardized Data:
        Age  Family size  latitude  longitude  Pin code
0 -1.557620     0.532929  0.102224  -0.016759 -1.247274
1 -0.211614    -0.208205  0.111227  -0.445712 -0.992164
```

- ## Data Transformation

  - In this step, various data transformation techniques are applied to the dataset. This could include techniques such as normalization, standardization, feature scaling, encoding categorical variables (like one-hot encoding), feature engineering (creating new features based on existing ones), and handling missing values (imputation, removal, etc.). These transformations aim to prepare the data for better performance of machine learning algorithms.

- ## Classification

  - Classification involves training machine learning models to predict the categorical target variable based on the features in the dataset. In this task, the following classification algorithms are applied:

  - K-Nearest Neighbors (KNN)
  - Support Vector Machine (SVM)
  - Naive Bayes
  - Logistic Regression
  - Decision Tree

Each algorithm is trained on both the original dataset and the pre-processed dataset to compare their performances.

- ## Evaluation:

  - In this step, the trained models are evaluated on both the original and pre-processed datasets.
  - Evaluation metrics such as confusion matrix and classification report are used to assess the performance of each model.
  - Confusion matrix provides a summary of correct and incorrect predictions made by the model, which helps in understanding the model's behavior in terms of true positives, false positives, true negatives, and false negatives.

  - Classification report provides a summary of precision, recall, F1-score, and support for each class, which gives insights into the overall performance of the model across different classes.
  - By comparing the evaluation results of models trained on the original and pre-processed datasets, the effectiveness of data preprocessing techniques can be analyzed.

PREPROCESSED DATA EVALUATION

```
KNN Classifier:
              precision    recall  f1-score   support

          No       0.62      0.42      0.50        12
         Yes       0.90      0.95      0.93        66

    accuracy                           0.87        78
   macro avg       0.76      0.69      0.71        78
weighted avg       0.86      0.87      0.86        78

Confusion Matrix for KNN Classifier:
[[ 5  7]
 [ 3 63]]

SVM Classifier:
              precision    recall  f1-score   support

          No       0.60      0.50      0.55        12
         Yes       0.91      0.94      0.93        66

    accuracy                           0.87        78
   macro avg       0.76      0.72      0.74        78
weighted avg       0.86      0.87      0.87        78

Confusion Matrix for SVM Classifier:
[[ 6  6]
 [ 4 62]]
```

ORIGINAL DATA EVALUATION

```
Evaluation on Original Data:

KNN Classifier:
              precision    recall  f1-score   support

          No       0.89      0.67      0.76        12
         Yes       0.94      0.98      0.96        66

    accuracy                           0.94        78
   macro avg       0.92      0.83      0.86        78
weighted avg       0.93      0.94      0.93        78

Confusion Matrix for KNN Classifier:
[[ 8  4]
 [ 1 65]]

SVM Classifier:
              precision    recall  f1-score   support

          No       0.00      0.00      0.00        12
         Yes       0.85      1.00      0.92        66

    accuracy                           0.85        78
   macro avg       0.42      0.50      0.46        78
weighted avg       0.72      0.85      0.78        78

Confusion Matrix for SVM Classifier:
[[ 0 12]
```

TASK 2

1. Introduction

In today's fiercely competitive retail industry, understanding customers' behavior and predicting churn is crucial for business success. Retailers need to identify their most profitable customer segments and implement effective strategies to retain them while also targeting potential high-value customers. This report aims to address the challenge of churn prediction and customer retention in the retail industry, focusing on a dataset provided by a food store in Pakistan spanning approximately 6 weeks.

2. Data Description

The dataset comprises three key columns:

- Visit_Date: Date when a customer visited the store.
- Customer_ID: Unique identifier for each customer.
- Total_Purchases_In_USD: Total sales made by the customer on a given day.

3. Preprocessing

3.1 Data Cleaning

- Handling missing values, duplicates, and outliers.
- Ensuring consistency and accuracy of data.

3.2 Feature Engineering

- Creating new features to enhance model performance, such as:
  - Total revenue before the reference date.
  - Maximum and minimum purchases in a single day.

- Total visit days in history.
- Standard deviation in purchase history.
- Weekly purchases and visit days before the reference date.
- Number of visits and total purchases on specific weekdays.
- Visit flags and sales amounts for days before the reference date.

4. Exploratory Data Analysis (EDA)

- Visualizing the distribution of key features.
- Analyzing correlations between features and churn status.
- Extracting insights to understand customer behavior and preferences.

5. Modeling Approach

- Selection of appropriate machine learning algorithms for churn prediction.
- Training and evaluation of models using relevant performance metrics.
- Fine-tuning model parameters to optimize performance.

6. Model Evaluation

- Assessing the performance of churn prediction models.
- Comparing predicted churn labels with actual labels.
- Calculating the percentage difference/error between rule-based and actual churn labels.

7. Results and Discussion

- Interpretation of model results and insights gained from the analysis.
- Identification of key factors influencing churn prediction.

	Visit_Date	CustomerID	Total_Purchases_In_USD	Day	Week	Days_Since_Last_Visit	Post_Period_Flag_x	W1_Total_Sales	W1_Visit_Days	W2_Total_Sales	...	D3_
0	2014-10-03	488	171.20	Friday	W3	NaN	False	171.20	1	171.20	...	
1	2014-10-09	5194	599.20	Thursday	W2	NaN	False	599.20	1	599.20	...	
2	2014-09-23	5398	51.36	Tuesday	W4	NaN	False	102.72	2	102.72	...	
3	2014-10-07	5398	51.36	Tuesday	W2	14.0	False	102.72	2	102.72	...	
4	2014-09-25	6930	85.60	Thursday	W4	NaN	False	599.20	6	599.20	...	

5 rows × 51 columns

.

TASK 2.2: EDA

- Calculate the week with the highest earning?

```
77] combined_data['Visit_Date'] = pd.to_datetime(combined_data['Visit_Date'])

    # Group by week and sum the Total_Purchases_In_USD for each week
    weekly_earnings = combined_data.groupby(combined_data['Visit_Date'].dt.isocalendar().week)['Total_Purchases_In_USD'].sum()

    # Find the week with the highest earnings
    highest_earning_week = weekly_earnings.idxmax()
    total_earnings_in_that_week = weekly_earnings.max()

    print("Week with the highest earnings:", highest_earning_week)
    print("Total earnings in that week:", total_earnings_in_that_week)
```

```
Week with the highest earnings: 40
Total earnings in that week: 28298666.9824
```

```
# Identify the most valued customer?

# Group by CustomerID and sum the Total_Purchases_In_USD for each customer
customer_total_purchases = combined_data.groupby('CustomerID')['Total_Purchases_In_USD'].sum()

# Find the customer with the highest total purchases
most_valued_customer = customer_total_purchases.idxmax()
highest_total_purchases = customer_total_purchases.max()

print("Most valued customer ID:", most_valued_customer)
print("Total purchases:", highest_total_purchases)
```

```
Most valued customer: 1032283346
Total purchases: 53414.4
```

- Identify the most valued customer?
- Categorize the customers into 3 groups (i.e., Poor, Mediocre, Rich)

```
9] rich_threshold = customer_total_purchases.quantile(0.67)

# Categorize customers based on their total purchases
def categorize_customer(total_purchases):
    if total_purchases <= poor_threshold:
        return 'Poor'
    elif total_purchases <= rich_threshold:
        return 'Mediocre'
    else:
        return 'Rich'

# Apply the categorization function to each customer
customer_categories = customer_total_purchases.apply(categorize_customer)

print(customer_categories.head())
```

```
CustomerID
488         Poor
5194    Mediocre
5398        Poor
6930    Mediocre
7260        Rich
Name: Total_Purchases_In_USD, dtype: object
```

- Will we able to figure out churn important factors from available data?

Ans: Yes, it's possible to identify important factors related to churn using the available data. Here are some common factors that can contribute to churn prediction:

- Frequency of Visits: Customers who visit less frequently are more likely to churn.
- Recent Activity: Customers who haven't made a purchase recently are more likely to churn.
- Total Purchases: Customers with lower total purchases might be more likely to churn.
- Day of Week: Analyzing which days of the week customers tend to make purchases or visit can provide insights into their behavior.
- Weekday vs. Weekend Behavior: Understanding if customers behave differently on weekdays versus weekends can also be informative.
- Customer Segmentation: Segmenting customers based on their behavior, demographics, or purchase history and analyzing churn within these segments can reveal patterns.
- Post-Period Flag: Whether a customer has made a purchase during the post-period can indicate potential churn.
- Days Since Last Visit: Customers who haven't visited in a long time may be more likely to churn.

By analyzing these factors and possibly others specific to your business context, you can build predictive models to identify customers at risk of churn and take proactive measures to retain them.

```python
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4

# Create and train the Decision Tree model
model = DecisionTreeClassifier()
model.fit(X_train, y_train)

# Evaluate the model
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Model accuracy: {accuracy:.2f}')

# Generate and print the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(conf_matrix)
```

```
Model accuracy: 1.00
Confusion Matrix:
[[131052]]
```

Accuracy on test data: 59%