Name - Sayyam Jain

PRN No – 22311031

Roll No – 282036

Batch – B2–SYB

**Assignment No – 2**

**Problem Statements :**

2. Perform the following operations using R/Python on the data sets:

a) Compute and display summary statistics for each feature available in the dataset. (e.g.

minimum value, maximum value, mean, range, standard deviation, variance and

percentiles

b) Illustrate the feature distributions using histogram.

c) Data cleaning, Data integration, Data transformation, Data model building (e.g.

Classification)

**Objective**  
Cleaning the data (removing extraneous columns, addressing missing values), transforming it (encoding categorical variables, scaling numerical features), and getting it ready for model creation by making sure it is consistent and compatible with machine learning techniques are all part of this assignment.

**Main Functions**

The code leverages pandas, numpy, and scikit-learn libraries. Key functions include:

1. import pandas as pd and import numpy as np: Import libraries for data manipulation and numerical operations.
2. pd.read\_csv('train.csv'): Loads the dataset into a pandas DataFrame.
3. dataset.describe(): Provides statistical summaries of numerical columns.
4. dataset.info(): Displays data types, non-null counts, and memory usage.
5. dataset.head(): Shows the first five rows for a quick preview.
6. dataset.isnull().sum(): Identifies missing values per column.
7. dataset.drop(columns=...): Removes specified columns (e.g., TransactionID, CardSecurityCode, PrimaryEmailDomain).
8. dataset.fillna(...): Imputes missing values (e.g., with the mean for numerical columns or specific values like "Unknown" for categorical columns).
9. sklearn.preprocessing.StandardScaler: Scales numerical features to have zero mean and unit variance.
10. sklearn.preprocessing.LabelEncoder: Encodes categorical variables into numerical labels.

**Methodology**

The code follows a systematic preprocessing pipeline:

1. Library Import: Import pandas, numpy, and scikit-learn tools (StandardScaler, LabelEncoder).
2. Data Loading: Load the dataset from train.csv into a DataFrame (dataset).
3. Exploratory Analysis:
   * Use describe() to summarize numerical columns (e.g., FraudLabel, D1, V283).
   * Use info() to check data types (111 float64, 2 int64, 13 object) and missing values.
   * Use head() to inspect the first few rows.
4. Data Cleaning:
   * Drop unnecessary columns (TransactionID, CardSecurityCode, PrimaryEmailDomain) deemed irrelevant.
   * Handle missing values:
     + Fill numerical columns with their mean (e.g., D1, D2).
     + Fill categorical columns (CardType, CardUsageType) with "Unknown".
5. Data Transformation:
   * Apply Standard Scaling to numerical columns using StandardScaler to normalize features.
   * Perform Label Encoding on categorical columns (e.g., ProductCategory, CardType) to convert them into numerical values.
6. Verification: Post-transformation checks with describe() and isnull().sum() ensure no missing values remain and scaling is applied correctly.

**Advantages**

* Comprehensive, easy-to-use preprocessing pipeline.
* Prepares data for machine learning with scaling and encoding.
* Efficient for moderate datasets (90,000 rows).
* Insightful exploration with pandas tools.

**Disadvantages**

* Basic missing value handling may bias results.
* LabelEncoder unsuitable for nominal data.
* No outlier detection or visualization.
* Drops potentially useful columns.
* Scaling assumes normality, risking issues with skewed data.

**Conclusion**

The train.csv dataset is efficiently cleaned, transformed, and standardized for machine learning by this code. It is excellent at creating a consistent dataset that is ready for modeling, but it is not very good at handling missing values, outliers, or categorical encoding. Consider including one-hot encoding for nominal categories, alternative imputation techniques, outlier identification, and visualizations to gain a better understanding of the data prior to modeling for a more robust process.