Perform preprocessing on Dataset

AIM:

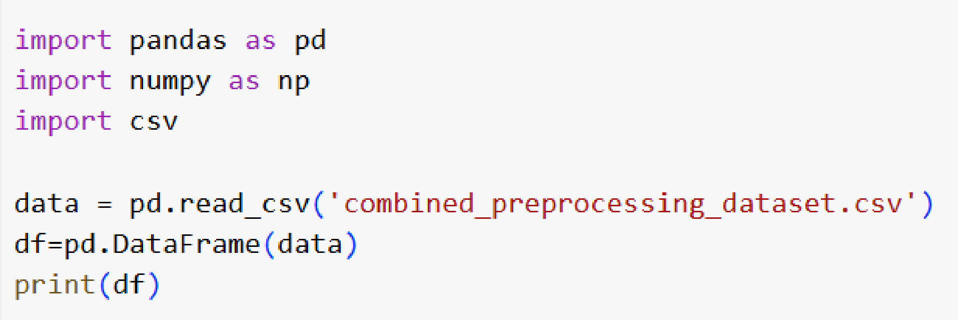
To preprocess a dataset by filling missing values, encoding categorical data, scaling and transforming features, detecting outliers, and selecting important features for effective machine learning.

PROCEDURE:

- Load the dataset combined_preprocessing_dataset.csv using pandas and convert it into a DataFrame.
- Handle missing values by filling null values in numerical columns like Age, Salary, and Income with their respective column means. For categorical columns like Department and Membership, fill missing values using the most frequent value (mode).
- Encode categorical data by applying one-hot encoding to the City column using pd.get_dummies() and label encoding to the Gender column using LabelEncoder.
- Perform feature scaling on selected numerical columns (Income, LoanAmount, Age) using both MinMaxScaler and StandardScaler from scikit-learn to demonstrate the effect of different scaling techniques.
- Detect outliers in the LoanAmount column using the Interquartile Range (IQR) method. Calculate Q1 and Q3, derive the IQR, and determine the lower and upper bounds to identify records that fall outside this range.
- Apply feature transformation by performing logarithmic transformation on Income and LoanAmount to reduce skewness. Scale the CreditScore column using RobustScaler to make it less sensitive to outliers.
- Perform feature selection by calculating the correlation matrix for selected features (Advertising, Price, Discount, Sales) and identifying the top 2 features most correlated with Sales based on absolute correlation values.

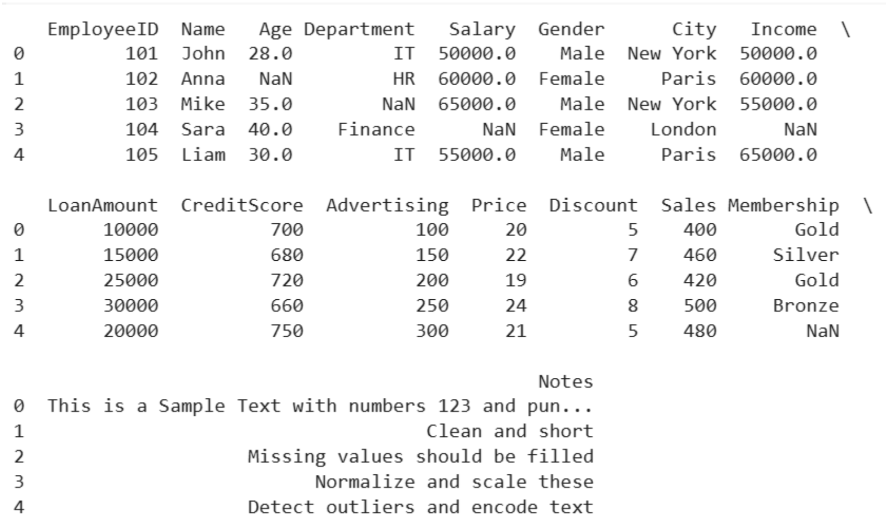
CODE:

Step 1: Import necessary packages and read the dataset into a DataFrame.



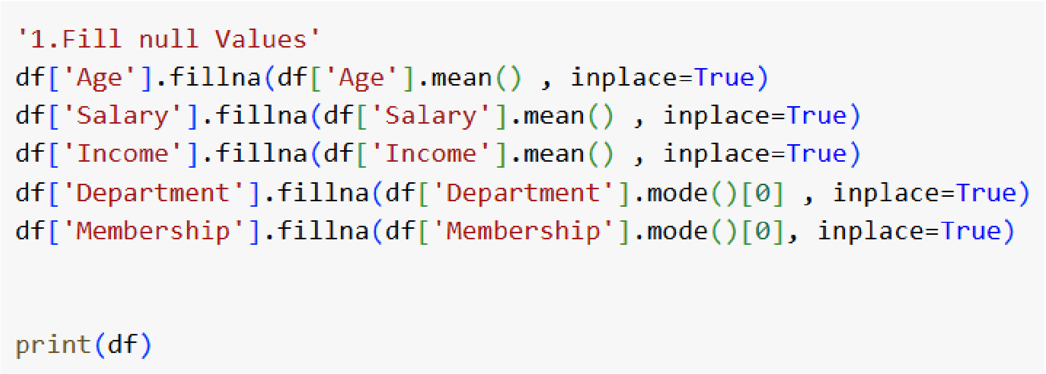
```
import pandas as pd
import numpy as np
import csv

data = pd.read_csv('combined_preprocessing_dataset.csv')
df=pd.DataFrame(data)
print(df)
```

OUTPUT:



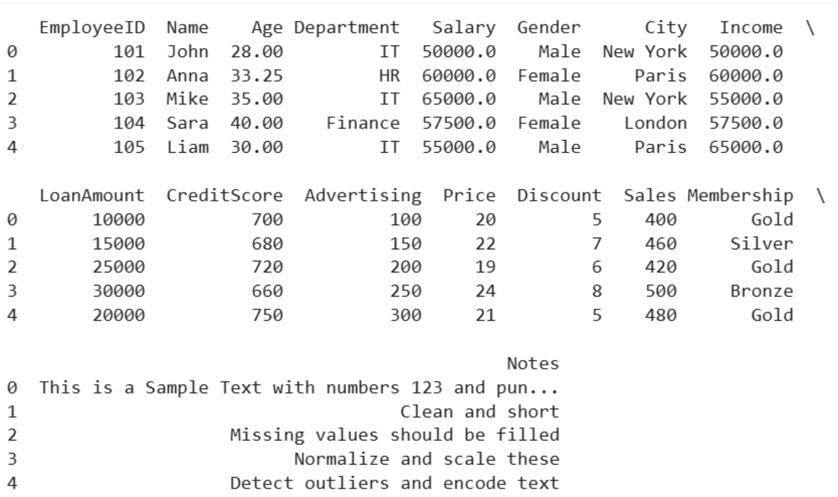
```
   EmployeeID  Name   Age Department   Salary  Gender      City   Income  \
0         101  John  28.0         IT  50000.0    Male  New York  50000.0   
1         102  Anna   NaN         HR  60000.0  Female     Paris  60000.0   
2         103  Mike  35.0        NaN  65000.0    Male  New York  55000.0   
3         104  Sara  40.0    Finance      NaN  Female    London      NaN   
4         105  Liam  30.0         IT  55000.0    Male     Paris  65000.0   

   LoanAmount  CreditScore  Advertising  Price  Discount  Sales Membership  \
0       10000          700          100     20         5    400       Gold   
1       15000          680          150     22         7    460     Silver   
2       25000          720          200     19         6    420       Gold   
3       30000          660          250     24         8    500     Bronze   
4       20000          750          300     21         5    480        NaN   

                                               Notes  
0  This is a Sample Text with numbers 123 and pun...  
1                                    Clean and short  
2                       Missing values should be filled  
3                          Normalize and scale these  
4                       Detect outliers and encode text  
```

Step 2: Handle Missing Values.



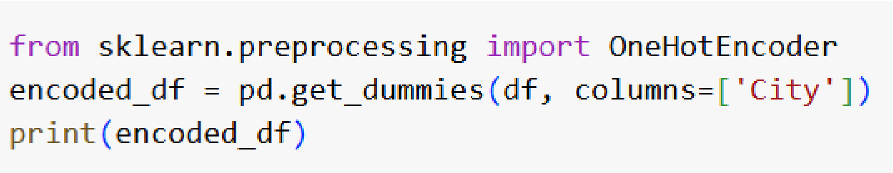
```
'1.Fill null Values'
df['Age'].fillna(df['Age'].mean() , inplace=True)
df['Salary'].fillna(df['Salary'].mean() , inplace=True)
df['Income'].fillna(df['Income'].mean() , inplace=True)
df['Department'].fillna(df['Department'].mode()[0] , inplace=True)
df['Membership'].fillna(df['Membership'].mode()[0], inplace=True)


print(df)
```

OUTPUT:



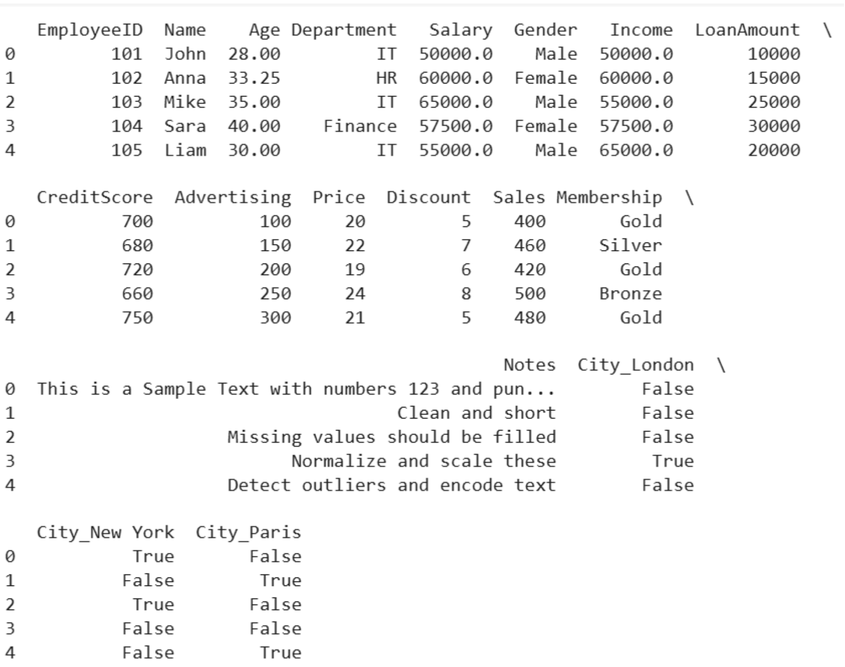
```
   EmployeeID  Name    Age Department   Salary  Gender      City   Income  \
0         101  John  28.00         IT  50000.0    Male  New York  50000.0   
1         102  Anna  33.25         HR  60000.0  Female     Paris  60000.0   
2         103  Mike  35.00         IT  65000.0    Male  New York  55000.0   
3         104  Sara  40.00    Finance  57500.0  Female    London  57500.0   
4         105  Liam  30.00         IT  55000.0    Male     Paris  65000.0   

   LoanAmount  CreditScore  Advertising  Price  Discount  Sales Membership  \
0       10000          700          100     20         5    400       Gold   
1       15000          680          150     22         7    460     Silver   
2       25000          720          200     19         6    420       Gold   
3       30000          660          250     24         8    500     Bronze   
4       20000          750          300     21         5    480       Gold   

                                               Notes  
0  This is a Sample Text with numbers 123 and pun...  
1                                    Clean and short  
2                       Missing values should be filled  
3                          Normalize and scale these  
4                       Detect outliers and encode text  
```

Step 3: Convert text data into numbers using one-hot encoding.



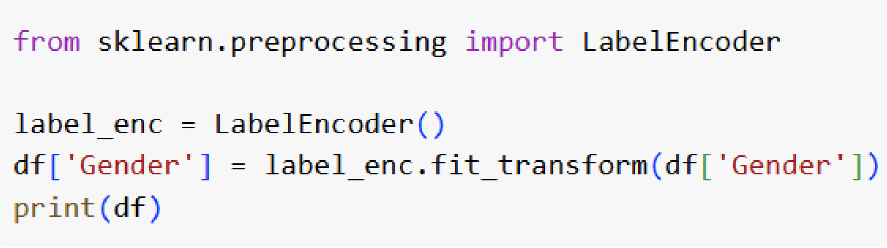
```
from sklearn.preprocessing import OneHotEncoder
encoded_df = pd.get_dummies(df, columns=['City'])
print(encoded_df)
```

OUTPUT:



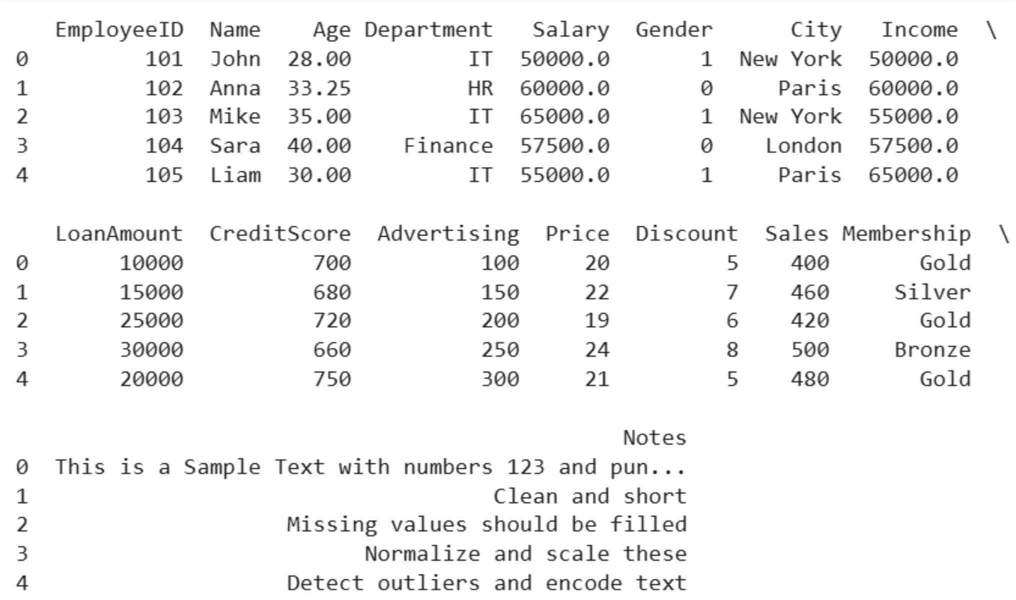
```
   EmployeeID  Name    Age Department   Salary  Gender   Income  LoanAmount  \
0         101  John  28.00         IT  50000.0    Male  50000.0       10000   
1         102  Anna  33.25         HR  60000.0  Female  60000.0       15000   
2         103  Mike  35.00         IT  65000.0    Male  55000.0       25000   
3         104  Sara  40.00    Finance  57500.0  Female  57500.0       30000   
4         105  Liam  30.00         IT  55000.0    Male  65000.0       20000   

   CreditScore  Advertising  Price  Discount  Sales Membership  \
0          700          100     20         5    400       Gold   
1          680          150     22         7    460     Silver   
2          720          200     19         6    420       Gold   
3          660          250     24         8    500     Bronze   
4          750          300     21         5    480       Gold   

                                          Notes  City_London  \
0  This is a Sample Text with numbers 123 and pun...        False   
1                                  Clean and short        False   
2                    Missing values should be filled        False   
3                          Normalize and scale these         True   
4                    Detect outliers and encode text        False   

   City_New York  City_Paris  
0           True       False  
1          False        True  
2           True       False  
3          False       False  
4          False        True  
```

Step 4: Convert text data into numbers using Label encoding.



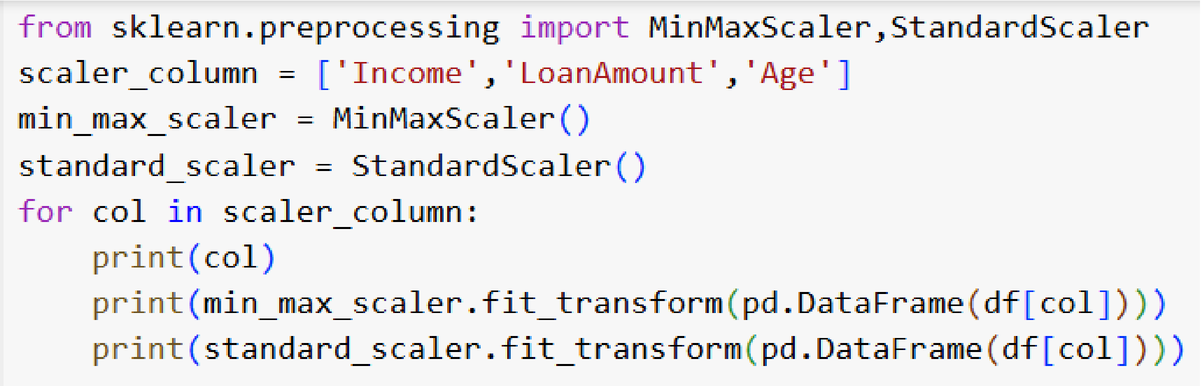
```python
from sklearn.preprocessing import LabelEncoder

label_enc = LabelEncoder()
df['Gender'] = label_enc.fit_transform(df['Gender'])
print(df)
```

OUTPUT:



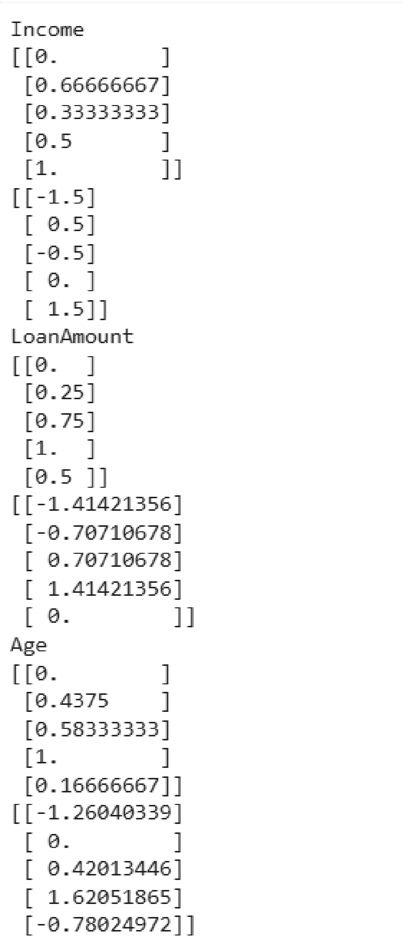
```
   EmployeeID  Name    Age Department   Salary  Gender      City   Income  \
0         101  John  28.00         IT  50000.0       1  New York  50000.0   
1         102  Anna  33.25         HR  60000.0       0     Paris  60000.0   
2         103  Mike  35.00         IT  65000.0       1  New York  55000.0   
3         104  Sara  40.00    Finance  57500.0       0    London  57500.0   
4         105  Liam  30.00         IT  55000.0       1     Paris  65000.0   

   LoanAmount  CreditScore  Advertising  Price  Discount  Sales Membership  \
0       10000          700          100     20         5    400       Gold   
1       15000          680          150     22         7    460     Silver   
2       25000          720          200     19         6    420       Gold   
3       30000          660          250     24         8    500     Bronze   
4       20000          750          300     21         5    480       Gold   

                                          Notes  
0  This is a Sample Text with numbers 123 and pun...  
1                                  Clean and short  
2                    Missing values should be filled  
3                          Normalize and scale these  
4                    Detect outliers and encode text  
```

Step 5: Normalize values using MinMax and Standard Scaler..



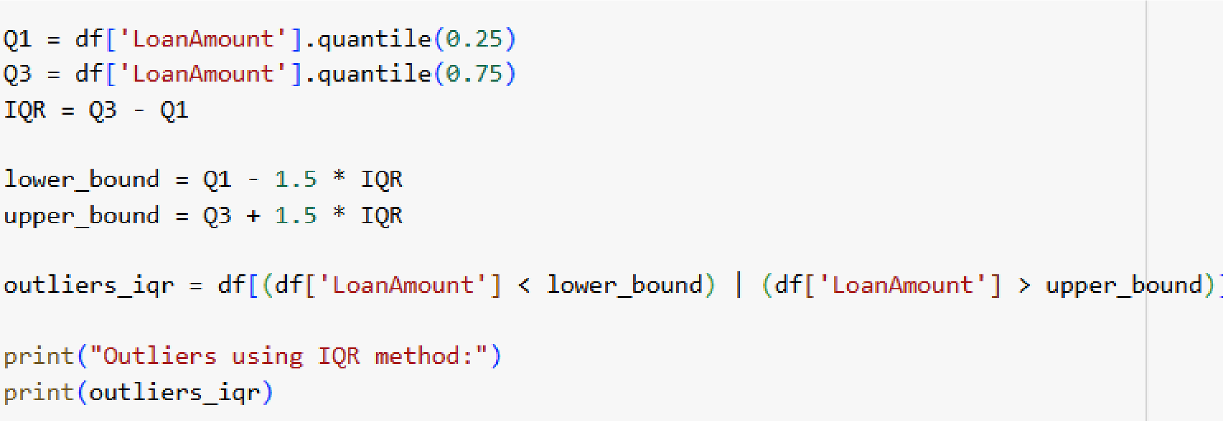
```
from sklearn.preprocessing import MinMaxScaler,StandardScaler
scaler_column = ['Income','LoanAmount','Age']
min_max_scaler = MinMaxScaler()
standard_scaler = StandardScaler()
for col in scaler_column:
    print(col)
    print(min_max_scaler.fit_transform(pd.DataFrame(df[col])))
    print(standard_scaler.fit_transform(pd.DataFrame(df[col])))
```

OUTPUT:



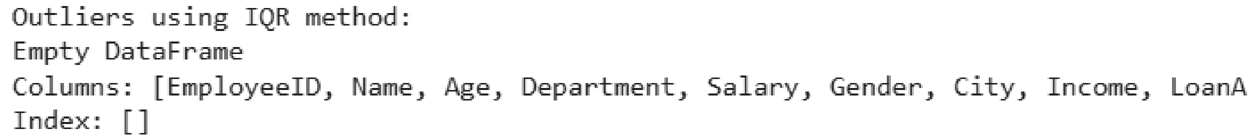
```
Income
[[0.        ]
 [0.66666667]
 [0.33333333]
 [0.5       ]
 [1.        ]]
[[-1.5]
 [ 0.5]
 [-0.5]
 [ 0. ]
 [ 1.5]]
LoanAmount
[[0.  ]
 [0.25]
 [0.75]
 [1.  ]
 [0.5 ]]
[[-1.41421356]
 [-0.70710678]
 [ 0.70710678]
 [ 1.41421356]
 [ 0.        ]]
Age
[[0.        ]
 [0.4375    ]
 [0.58333333]
 [1.        ]
 [0.16666667]]
[[-1.26040339]
 [ 0.        ]
 [ 0.42013446]
 [ 1.62051865]
 [-0.78024972]]
```

Step 6: Identify Outlier values using the IQR method.



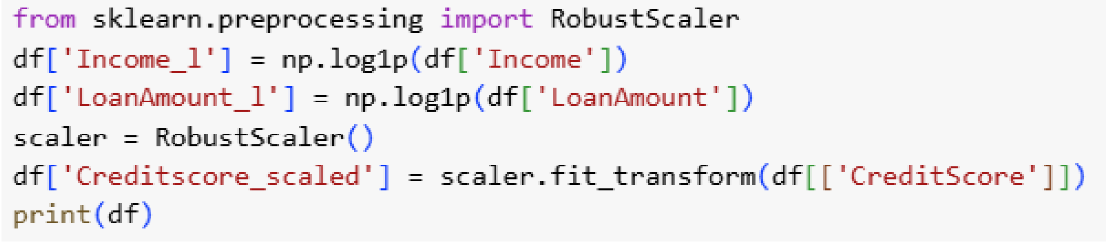
```
Q1 = df['LoanAmount'].quantile(0.25)
Q3 = df['LoanAmount'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers_iqr = df[(df['LoanAmount'] < lower_bound) | (df['LoanAmount'] > upper_bound)]

print("Outliers using IQR method:")
print(outliers_iqr)
```

OUTPUT:



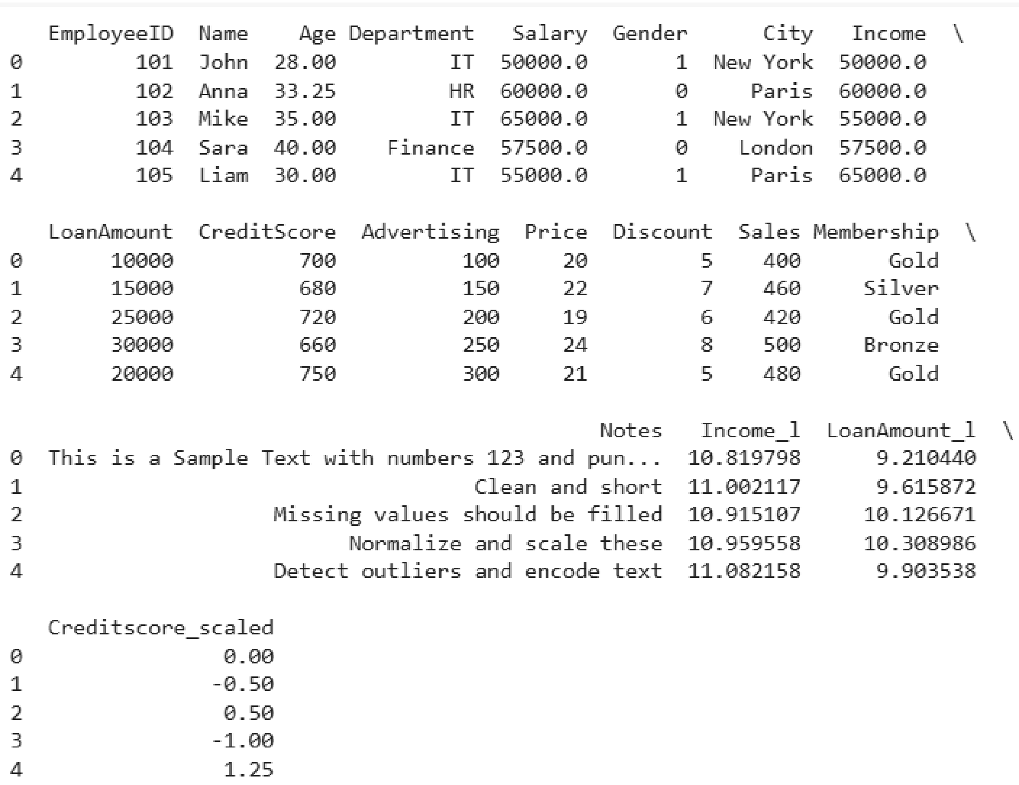
```
Outliers using IQR method:
Empty DataFrame
Columns: [EmployeeID, Name, Age, Department, Salary, Gender, City, Income, LoanA
Index: []
```

Step 7: Feature Transformation.



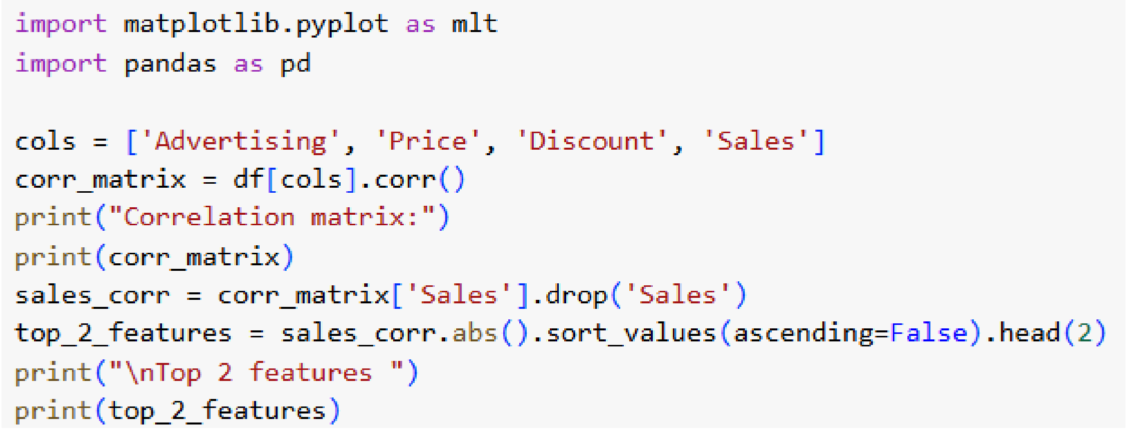
```python
from sklearn.preprocessing import RobustScaler
df['Income_l'] = np.log1p(df['Income'])
df['LoanAmount_l'] = np.log1p(df['LoanAmount'])
scaler = RobustScaler()
df['Creditscore_scaled'] = scaler.fit_transform(df[['CreditScore']])
print(df)
```

OUTPUT:



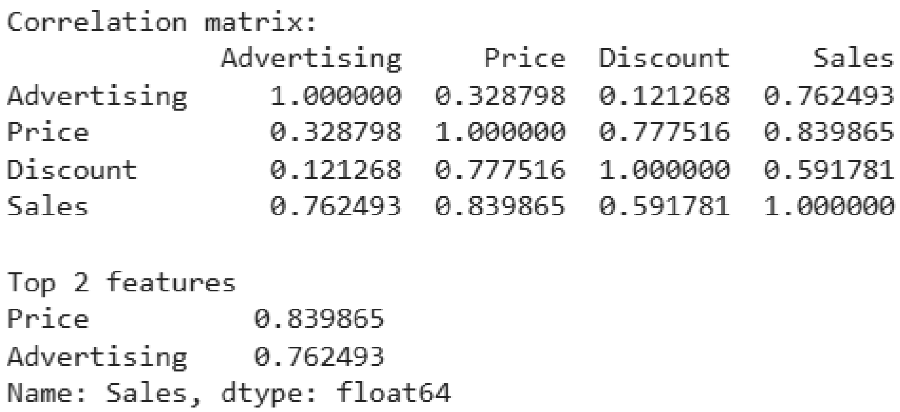
```
   EmployeeID  Name    Age Department   Salary  Gender      City   Income  \
0         101  John  28.00         IT  50000.0       1  New York  50000.0   
1         102  Anna  33.25         HR  60000.0       0     Paris  60000.0   
2         103  Mike  35.00         IT  65000.0       1  New York  55000.0   
3         104  Sara  40.00    Finance  57500.0       0    London  57500.0   
4         105  Liam  30.00         IT  55000.0       1     Paris  65000.0   

   LoanAmount  CreditScore  Advertising  Price  Discount  Sales Membership  \
0       10000          700          100     20         5    400       Gold   
1       15000          680          150     22         7    460     Silver   
2       25000          720          200     19         6    420       Gold   
3       30000          660          250     24         8    500     Bronze   
4       20000          750          300     21         5    480       Gold   

                                               Notes   Income_l  LoanAmount_l  \
0  This is a Sample Text with numbers 123 and pun...  10.819798      9.210440   
1                                    Clean and short  11.002117      9.615872   
2                         Missing values should be filled  10.915107     10.126671   
3                             Normalize and scale these  10.959558     10.308986   
4                         Detect outliers and encode text  11.082158      9.903538   

   Creditscore_scaled  
0                0.00  
1               -0.50  
2                0.50  
3               -1.00  
4                1.25  
```

Step 8: Feature Selection Using Correlation.



```python
import matplotlib.pyplot as mlt
import pandas as pd

cols = ['Advertising', 'Price', 'Discount', 'Sales']
corr_matrix = df[cols].corr()
print("Correlation matrix:")
print(corr_matrix)
sales_corr = corr_matrix['Sales'].drop('Sales')
top_2_features = sales_corr.abs().sort_values(ascending=False).head(2)
print("\nTop 2 features ")
print(top_2_features)
```

OUTPUT:



```
Correlation matrix:
             Advertising     Price  Discount     Sales
Advertising     1.000000  0.328798  0.121268  0.762493
Price           0.328798  1.000000  0.777516  0.839865
Discount        0.121268  0.777516  1.000000  0.591781
Sales           0.762493  0.839865  0.591781  1.000000

Top 2 features
Price          0.839865
Advertising    0.762493
Name: Sales, dtype: float64
```

RESULT:

Thus, data preprocessing with cleaning, encoding, scaling, outlier detection, and feature selection is executed successfully.