```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import RobustScaler
import seaborn as sns
from pandas.plotting import scatter_matrix
from sklearn_extra.cluster import KMedoids
from scipy.cluster.hierarchy import dendrogram, linkage

df = pd.read_csv(r"CC GENERAL.csv")
df
```

```
     CUST_ID      BALANCE  BALANCE_FREQUENCY  PURCHASES
ONEOFF_PURCHASES  \
0     C10001    40.900749           0.818182      95.40
0.00
1     C10002  3202.467416           0.909091       0.00
0.00
2     C10003  2495.148862           1.000000     773.17
773.17
3     C10004  1666.670542           0.636364    1499.00
1499.00
4     C10005   817.714335           1.000000      16.00
16.00
...      ...          ...                ...        ...
...
8945  C19186    28.493517           1.000000     291.12
0.00
8946  C19187    19.183215           1.000000     300.00
0.00
8947  C19188    23.398673           0.833333     144.40
0.00
8948  C19189    13.457564           0.833333       0.00
0.00
8949  C19190   372.708075           0.666667    1093.25
1093.25

      INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
0                      95.40      0.000000             0.166667
1                       0.00   6442.945483             0.000000
2                       0.00      0.000000             1.000000
3                       0.00    205.788017             0.083333
4                       0.00      0.000000             0.083333
...                      ...           ...                  ...
8945                  291.12      0.000000             1.000000
8946                  300.00      0.000000             1.000000
8947                  144.40      0.000000             0.833333
8948                    0.00     36.558778             0.000000
8949                    0.00    127.040008             0.666667
```

```
      ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
0                       0.000000                          0.083333
1                       0.000000                          0.000000
2                       1.000000                          0.000000
3                       0.083333                          0.000000
4                       0.083333                          0.000000
...                          ...                               ...
8945                    0.000000                          0.833333
8946                    0.000000                          0.833333
8947                    0.000000                          0.666667
8948                    0.000000                          0.000000
8949                    0.666667                          0.000000

      CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX
CREDIT_LIMIT  \
0                   0.000000                 0              2
1000.0
1                   0.250000                 4              0
7000.0
2                   0.000000                 0             12
7500.0
3                   0.083333                 1              1
7500.0
4                   0.000000                 0              1
1200.0
...                      ...               ...            ...
...
8945                0.000000                 0              6
1000.0
8946                0.000000                 0              6
1000.0
8947                0.000000                 0              5
1000.0
8948                0.166667                 2              0
500.0
8949                0.333333                 2             23
1200.0

          PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT  TENURE
0      201.802084        139.509787          0.000000      12
1     4103.032597       1072.340217          0.222222      12
2      622.066742        627.284787          0.000000      12
3        0.000000               NaN          0.000000      12
4      678.334763        244.791237          0.000000      12
...           ...               ...               ...     ...
8945   325.594462         48.886365          0.500000       6
8946   275.861322               NaN          0.000000       6
8947    81.270775         82.418369          0.250000       6
8948    52.549959         55.755628          0.250000       6
```

```
8949    63.165404          88.288956           0.000000        6

[8950 rows x 18 columns]
```

```r
# Data Preprocessing:
# Display sum of missing values before cleaning
print("Sum of NA before cleaning:")
print(df.isna().sum())

# Drop rows with missing values
df_cleaned = df.dropna()

# Display sum of missing values after cleaning
print("\nSum of NA after cleaning:")
print(df_cleaned.isna().sum())

# Check for duplicates
print("\nDuplicates before removal:")
print(df_cleaned.duplicated().sum())

# Remove duplicates
df_no_duplicates = df_cleaned.drop_duplicates()

# Get numerical columns
num_cols = df_no_duplicates.select_dtypes(include=np.number).columns

# Handle Outliers
z_scores = np.abs((df_no_duplicates[num_cols] -
df_no_duplicates[num_cols].mean()) / df_no_duplicates[num_cols].std())
outliers = z_scores > 3

# Replace outliers with NaN
df_no_duplicates.loc[outliers.any(axis=1), num_cols] = np.nan
df_no_duplicates = df_no_duplicates.dropna()

# Data Normalization or Scaling
scaler = RobustScaler()
df_no_duplicates[num_cols] =
scaler.fit_transform(df_no_duplicates[num_cols])
```

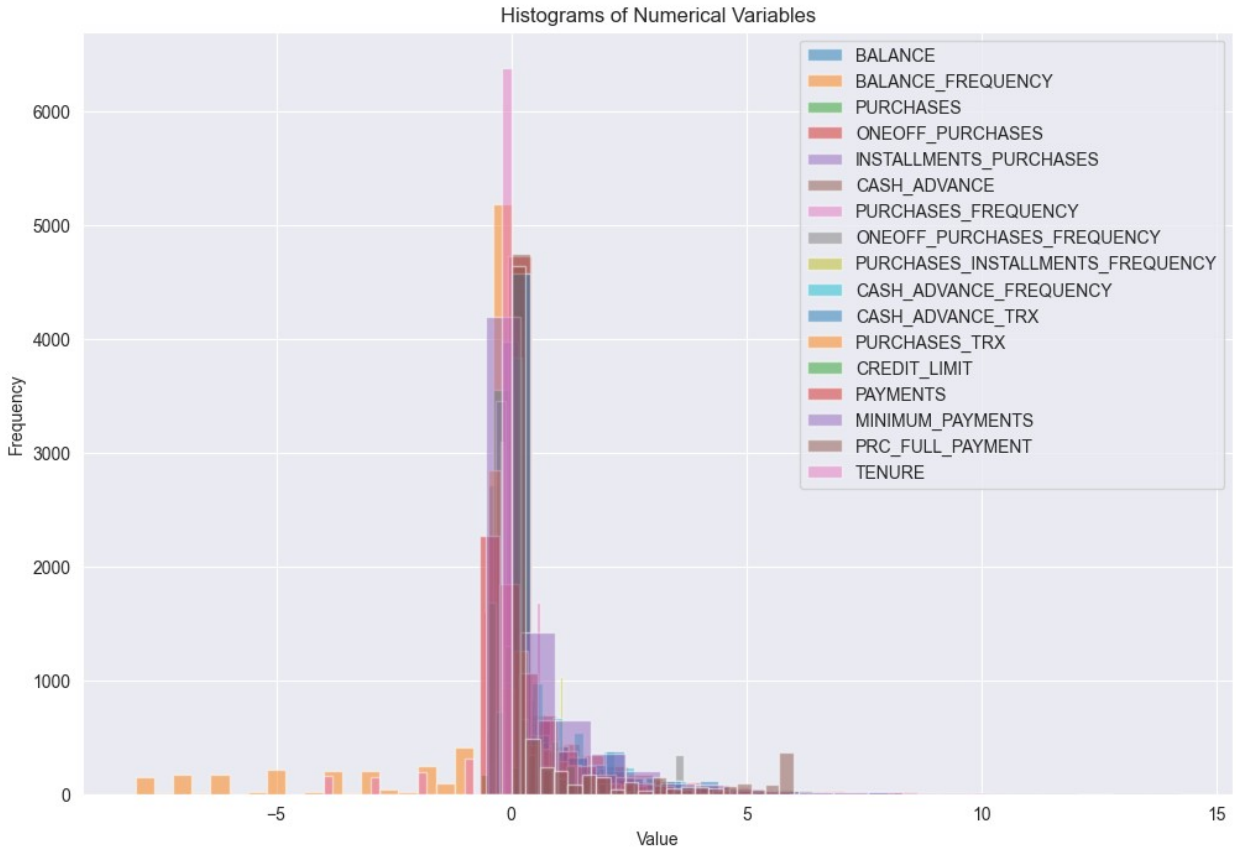
```
Sum of NA before cleaning:
CUST_ID                             0
BALANCE                             0
BALANCE_FREQUENCY                   0
PURCHASES                           0
ONEOFF_PURCHASES                    0
INSTALLMENTS_PURCHASES              0
CASH_ADVANCE                        0
PURCHASES_FREQUENCY                 0
ONEOFF_PURCHASES_FREQUENCY          0
PURCHASES_INSTALLMENTS_FREQUENCY    0
```

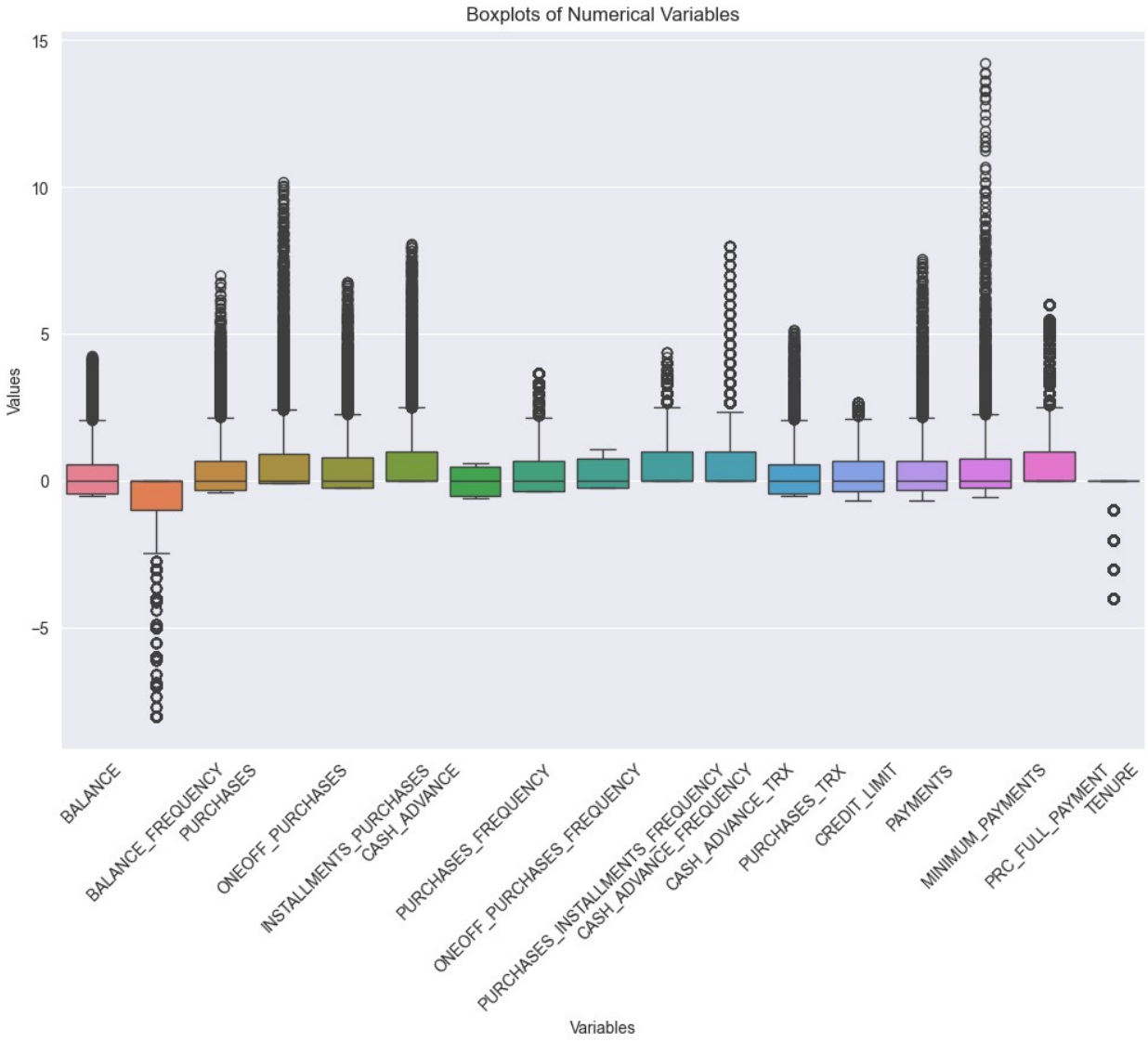
```
CASH_ADVANCE_FREQUENCY                0
CASH_ADVANCE_TRX                      0
PURCHASES_TRX                         0
CREDIT_LIMIT                          1
PAYMENTS                              0
MINIMUM_PAYMENTS                    313
PRC_FULL_PAYMENT                      0
TENURE                                0
dtype: int64

Sum of NA after cleaning:
CUST_ID                               0
BALANCE                               0
BALANCE_FREQUENCY                     0
PURCHASES                             0
ONEOFF_PURCHASES                      0
INSTALLMENTS_PURCHASES                0
CASH_ADVANCE                          0
PURCHASES_FREQUENCY                   0
ONEOFF_PURCHASES_FREQUENCY            0
PURCHASES_INSTALLMENTS_FREQUENCY      0
CASH_ADVANCE_FREQUENCY                0
CASH_ADVANCE_TRX                      0
PURCHASES_TRX                         0
CREDIT_LIMIT                          0
PAYMENTS                              0
MINIMUM_PAYMENTS                      0
PRC_FULL_PAYMENT                      0
TENURE                                0
dtype: int64

Duplicates before removal:
0

#  Exploratory Data Analysis (EDA):
# Display basic statistical summaries
print("Statistical Summaries:")
print(df_no_duplicates.describe())

# Plot histograms for numerical columns
plt.figure(figsize=(12, 8))
for column in num_cols:
    plt.hist(df_no_duplicates[column], bins=20, alpha=0.5,
label=column)
plt.title("Histograms of Numerical Variables")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.legend()
plt.show()
```

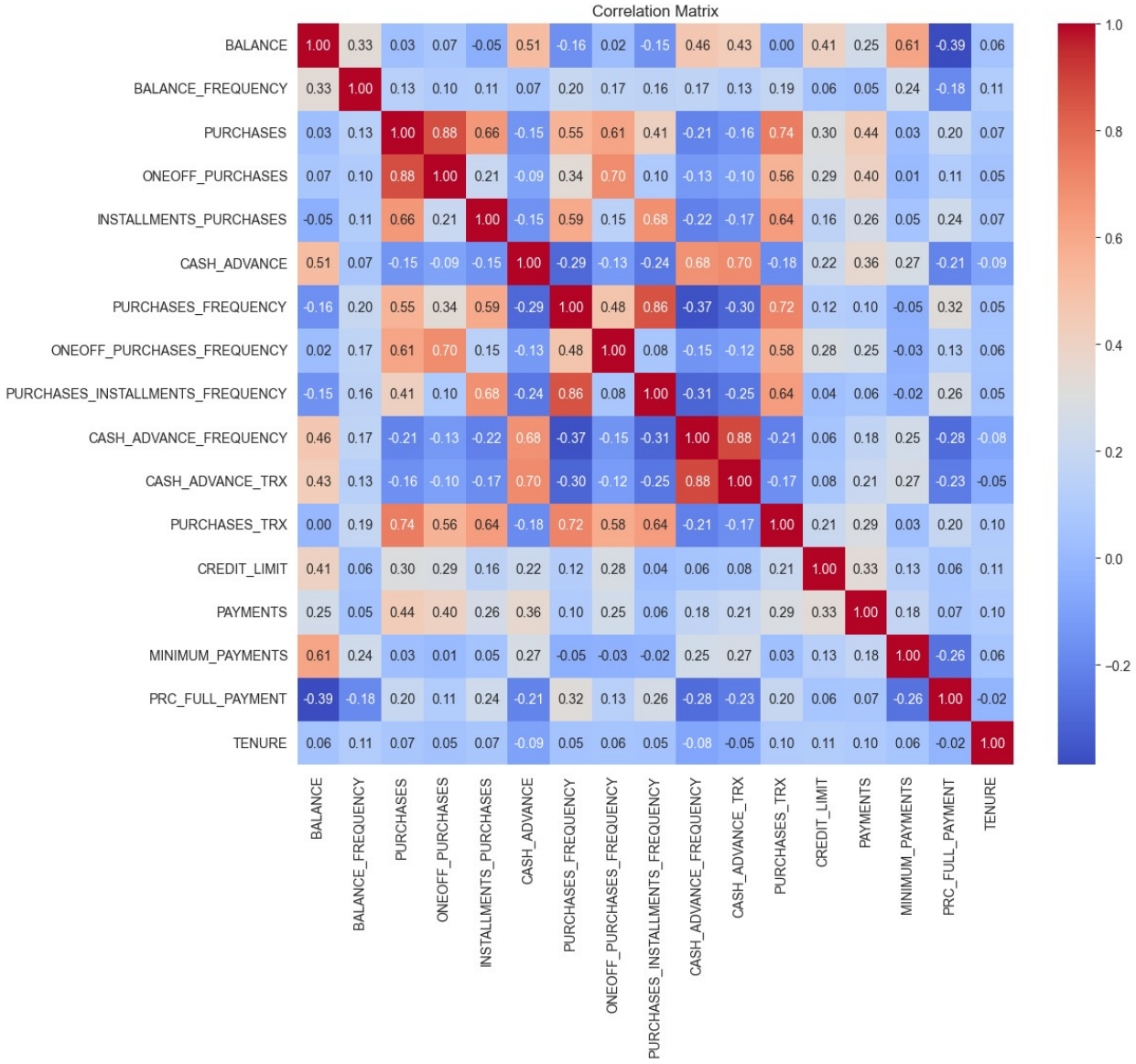
```
# Create Box-plots for numerical columns
plt.figure(figsize=(12, 8))
sns.boxplot(data=df_no_duplicates[num_cols])
plt.title("Boxplots of Numerical Variables")
plt.xlabel("Variables")
plt.ylabel("Values")
plt.xticks(rotation=45)
plt.show()

# Calculate correlation matrix
correlation_matrix = df_no_duplicates[num_cols].corr()

# Create heatmap for correlation matrix
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm',
fmt=".2f")
plt.title("Correlation Matrix")
plt.show()

numerical_columns_for_scatter_matrix = ['BALANCE', 'PURCHASES',
'ONEOFF_PURCHASES',
                     'INSTALLMENTS_PURCHASES', 'CASH_ADVANCE',
                     'CREDIT_LIMIT', 'PAYMENTS', 'MINIMUM_PAYMENTS']

# Create scatter matrix
scatter_matrix(df_no_duplicates[numerical_columns_for_scatter_matrix],
figsize=(12, 12), alpha=0.5)
plt.suptitle("Scatter Matrix of Numerical Variables")
plt.show()
```

```
Statistical Summaries:
            BALANCE  BALANCE_FREQUENCY    PURCHASES  ONEOFF_PURCHASES
\
count  7.190000e+03        7190.000000  7190.000000      7.190000e+03

mean   2.844004e-01          -1.032095     0.397023      7.456391e-01

std    9.222222e-01           2.054717     1.045919      1.494215e+00

min   -5.008548e-01          -8.000011    -0.376943     -7.131493e-02

25%   -4.196008e-01          -1.000000    -0.326808     -7.131493e-02

50%   -3.417947e-17           0.000000     0.000000     -6.613633e-18

75%    5.803992e-01           0.000000     0.673192      9.286851e-01

max    4.244249e+00           0.000000     6.986845      1.015754e+01


       INSTALLMENTS_PURCHASES  CASH_ADVANCE  PURCHASES_FREQUENCY  \
```

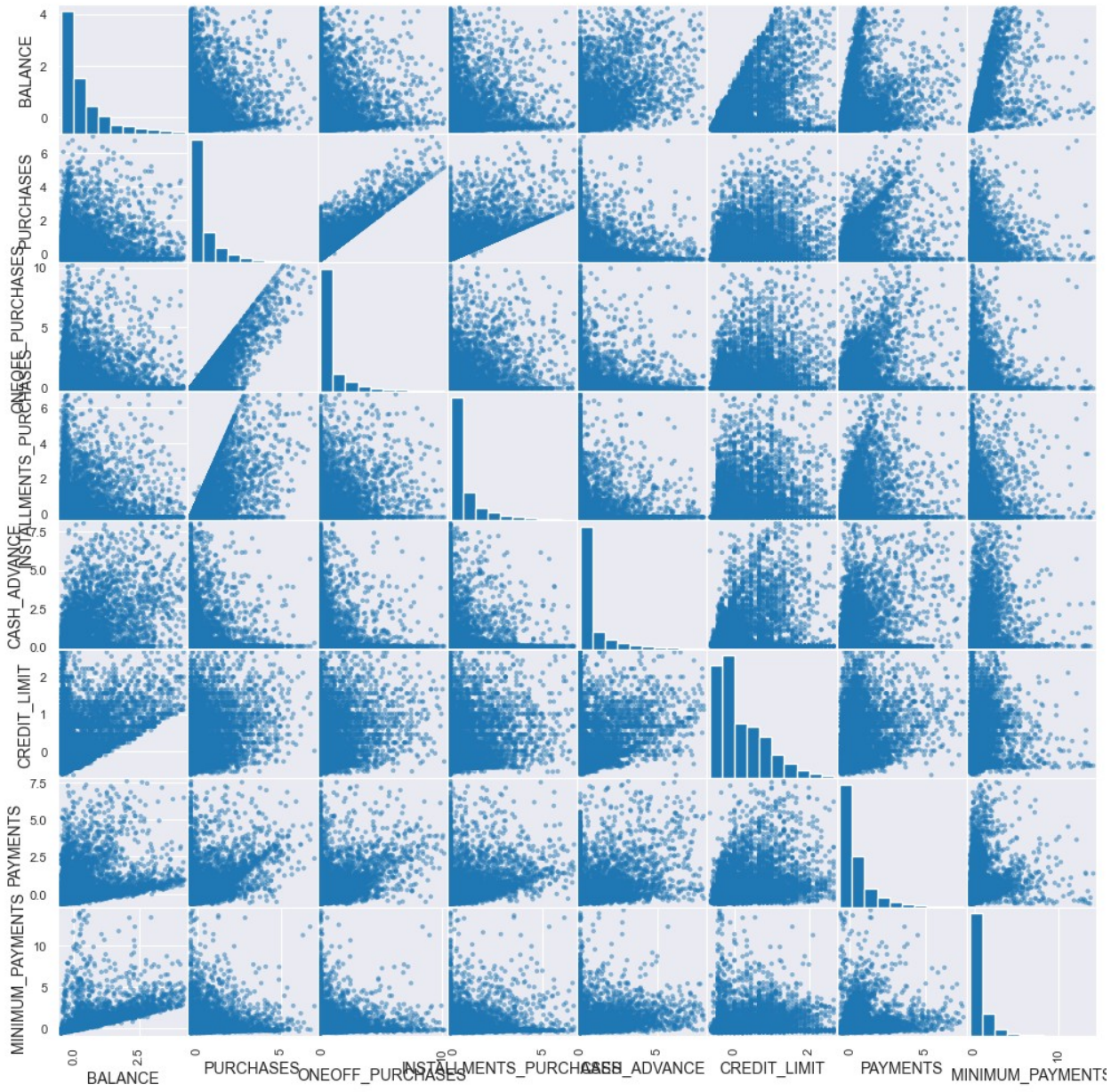
```
count             7.190000e+03   7190.000000           7190.000000
mean              5.161220e-01      0.756310             -0.007234
std               1.134427e+00      1.379168              0.475888
min              -2.134256e-01      0.000000             -0.600000
25%              -2.134256e-01      0.000000             -0.500000
50%               1.566672e-17      0.000000              0.000000
75%               7.865744e-01      1.000000              0.500000
max               6.765165e+00      8.051039              0.600000

       ONEOFF_PURCHASES_FREQUENCY  PURCHASES_INSTALLMENTS_FREQUENCY  \
count                 7190.000000                      7190.000000
mean                     0.447775                         0.244202
std                      1.157865                         0.525371
min                     -0.333332                        -0.242424
25%                     -0.333332                        -0.242424
50%                      0.000000                         0.000000
75%                      0.666668                         0.757576
max                      3.666668                         1.090909

       CASH_ADVANCE_FREQUENCY  CASH_ADVANCE_TRX  PURCHASES_TRX
CREDIT_LIMIT  \
count             7190.000000       7190.000000    7190.000000
7190.000000
mean                 0.676431          0.798285       0.272288
0.227690
std                  0.966767          1.354620       0.958378
0.657528
min                  0.000000          0.000000      -0.500000     -
0.655556
25%                  0.000000          0.000000      -0.437500     -
0.333333
50%                  0.000000          0.000000       0.000000
0.000000
75%                  1.000000          1.000000       0.562500
0.666667
max                  4.363629          8.000000       5.125000
2.666667

          PAYMENTS  MINIMUM_PAYMENTS  PRC_FULL_PAYMENT       TENURE
count  7190.000000       7190.000000       7190.000000  7190.000000
mean      0.404249          0.564362          0.937632    -0.246592
std       1.159387          1.471924          1.746280     0.793087
min      -0.666263         -0.538101          0.000000    -4.000000
25%      -0.331089         -0.223892          0.000000     0.000000
50%       0.000000          0.000000          0.000000     0.000000
75%       0.668911          0.776108          1.000000     0.000000
max       7.557902         14.234698          5.999988     0.000000
```


Histograms of Numerical Variables
BALANCE
BALANCE_FREQUENCY
PURCHASES
ONEOFF_PURCHASES
INSTALLMENTS_PURCHASES
CASH_ADVANCE
PURCHASES_FREQUENCY
ONEOFF_PURCHASES_FREQUENCY
PURCHASES_INSTALLMENTS_FREQUENCY
CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
PURCHASES_TRX
CREDIT_LIMIT
PAYMENTS
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
TENURE
Frequency
0
1000
2000
3000
4000
5000
6000
−5
0
5
10
15
Value


Boxplots of Numerical Variables
Values
15
10
5
0
−5
BALANCE
BALANCE_FREQUENCY
PURCHASES
ONEOFF_PURCHASES
INSTALLMENTS_PURCHASES
CASH_ADVANCE
PURCHASES_FREQUENCY
ONEOFF_PURCHASES_FREQUENCY
PURCHASES_INSTALLMENTS_FREQUENCY
CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
PURCHASES_TRX
CREDIT_LIMIT
PAYMENTS
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
TENURE
Variables


Correlation Matrix
BALANCE 1.00 0.33 0.03 0.07 -0.05 0.51 -0.16 0.02 -0.15 0.46 0.43 0.00 0.41 0.25 0.61 -0.39 0.06
BALANCE_FREQUENCY 0.33 1.00 0.13 0.10 0.11 0.07 0.20 0.17 0.16 0.17 0.13 0.19 0.06 0.05 0.24 -0.18 0.11
PURCHASES 0.03 0.13 1.00 0.88 0.66 -0.15 0.55 0.61 0.41 -0.21 -0.16 0.74 0.30 0.44 0.03 0.20 0.07
ONEOFF_PURCHASES 0.07 0.10 0.88 1.00 0.21 -0.09 0.34 0.70 0.10 -0.13 -0.10 0.56 0.29 0.40 0.01 0.11 0.05
INSTALLMENTS_PURCHASES -0.05 0.11 0.66 0.21 1.00 -0.15 0.59 0.15 0.68 -0.22 -0.17 0.64 0.16 0.26 0.05 0.24 0.07
CASH_ADVANCE 0.51 0.07 -0.15 -0.09 -0.15 1.00 -0.29 -0.13 -0.24 0.68 0.70 -0.18 0.22 0.36 0.27 -0.21 -0.09
PURCHASES_FREQUENCY -0.16 0.20 0.55 0.34 0.59 -0.29 1.00 0.48 0.86 -0.37 -0.30 0.72 0.12 0.10 -0.05 0.32 0.05
ONEOFF_PURCHASES_FREQUENCY 0.02 0.17 0.61 0.70 0.15 -0.13 0.48 1.00 0.08 -0.15 -0.12 0.58 0.28 0.25 -0.03 0.13 0.06
PURCHASES_INSTALLMENTS_FREQUENCY -0.15 0.16 0.41 0.10 0.68 -0.24 0.86 0.08 1.00 -0.31 -0.25 0.64 0.04 0.06 -0.02 0.26 0.05
CASH_ADVANCE_FREQUENCY 0.46 0.17 -0.21 -0.13 -0.22 0.68 -0.37 -0.15 -0.31 1.00 0.88 -0.21 0.06 0.18 0.25 -0.28 -0.08
CASH_ADVANCE_TRX 0.43 0.13 -0.16 -0.10 -0.17 0.70 -0.30 -0.12 -0.25 0.88 1.00 -0.17 0.08 0.21 0.27 -0.23 -0.05
PURCHASES_TRX 0.00 0.19 0.74 0.56 0.64 -0.18 0.72 0.58 0.64 -0.21 -0.17 1.00 0.21 0.29 0.03 0.20 0.10
CREDIT_LIMIT 0.41 0.06 0.30 0.29 0.16 0.22 0.12 0.28 0.04 0.06 0.08 0.21 1.00 0.33 0.13 0.06 0.11
PAYMENTS 0.25 0.05 0.44 0.40 0.26 0.36 0.10 0.25 0.06 0.18 0.21 0.29 0.33 1.00 0.18 0.07 0.10
MINIMUM_PAYMENTS 0.61 0.24 0.03 0.01 0.05 0.27 -0.05 -0.03 -0.02 0.25 0.27 0.03 0.13 0.18 1.00 -0.26 0.06
PRC_FULL_PAYMENT -0.39 -0.18 0.20 0.11 0.24 -0.21 0.32 0.13 0.26 -0.28 -0.23 0.20 0.06 0.07 -0.26 1.00 -0.02
TENURE 0.06 0.11 0.07 0.05 0.07 -0.09 0.05 0.06 0.05 -0.08 -0.05 0.10 0.11 0.10 0.06 -0.02 1.00
BALANCE
BALANCE_FREQUENCY
PURCHASES
ONEOFF_PURCHASES
INSTALLMENTS_PURCHASES
CASH_ADVANCE
PURCHASES_FREQUENCY
ONEOFF_PURCHASES_FREQUENCY
PURCHASES_INSTALLMENTS_FREQUENCY
CASH_ADVANCE_FREQUENCY
CASH_ADVANCE_TRX
PURCHASES_TRX
CREDIT_LIMIT
PAYMENTS
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
TENURE
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

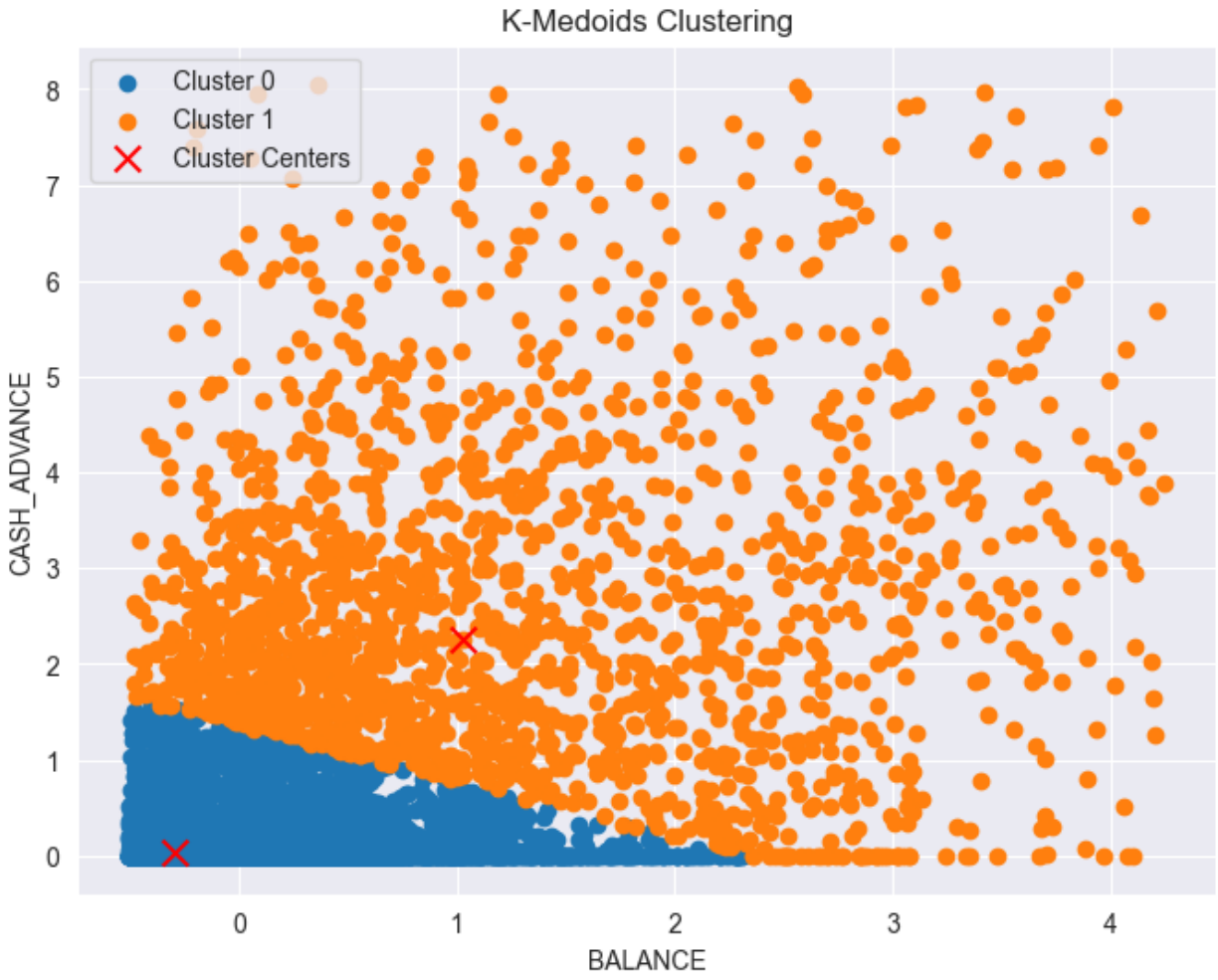
Scatter Matrix of Numerical Variables



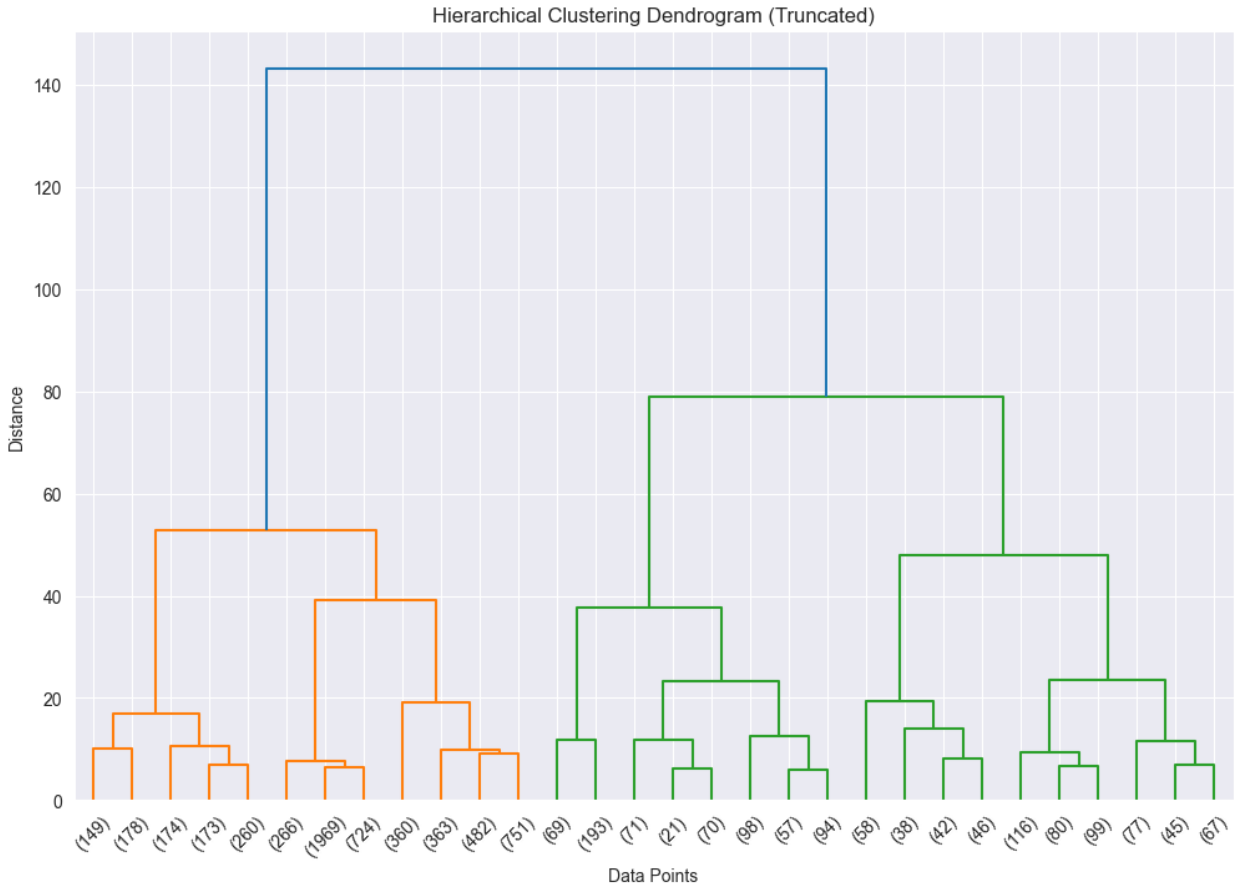

```
# Convert the selected columns to a NumPy array
selected_columns = ['BALANCE', 'CASH_ADVANCE']
X = df_no_duplicates[selected_columns]
data = np.array(X)

# K-Medoids clustering algorithm
k = 2
kmedoids = KMedoids(n_clusters=k).fit(data)
```

```
clusters = kmedoids.cluster_centers_
labels = kmedoids.labels_
print("Labels:", labels, "\n")
print("Clusters:\n", clusters, "\n")

for j in range(k):
    print("Cluster", j, ":", data[labels == j])

# Plotting the clusters
plt.figure(figsize=(8, 6))

# Plotting points for each cluster
for j in range(k):
    cluster_points = data[labels == j]
    plt.scatter(cluster_points[:, 0], cluster_points[:, 1],
label=f'Cluster {j}')

# Plotting cluster centers
plt.scatter(clusters[:, 0], clusters[:, 1], marker='x', color='red',
s=100, label='Cluster Centers')
plt.xlabel(selected_columns[0])
plt.ylabel(selected_columns[1])
plt.title('K-Medoids Clustering')
plt.legend()
plt.grid(True)
plt.show()
```

```
Labels: [0 1 0 ... 0 0 0]

Clusters:
 [[-0.29099562  0.03053921]
 [ 1.02431114  2.26441642]]

Cluster 0 : [[-0.47624993  0.        ]
 [ 1.00033835  0.        ]
 [-0.00888323  0.        ]
 ...
 [ 0.02217009  1.27034567]
 [-0.41798405  0.        ]
 [-0.30167071  0.46164909]]
Cluster 1 : [[ 1.42589365  7.09297957]
 [ 3.64220417  2.53369062]
 [ 0.74579732  3.06518248]
 ...
 [-0.01756237  2.81840998]
 [-0.0875243   1.92535437]
 [ 0.33118024  4.57859463]]
```




```
# Perform hierarchical clustering:
linkage_matrix = linkage(data, method='ward', metric='euclidean')

# Plot the dendrogram without the last node
plt.figure(figsize=(12, 8))
dendrogram(linkage_matrix, truncate_mode='lastp', p=30)
plt.title('Hierarchical Clustering Dendrogram (Truncated)')
plt.xlabel('Data Points')
plt.ylabel('Distance')
plt.show()
```


Hierarchical Clustering Dendrogram (Truncated)
Distance
0
20
40
60
80
100
120
140
(149)
(178)
(174)
(173)
(260)
(266)
(1969)
(724)
(360)
(363)
(482)
(751)
(69)
(193)
(71)
(21)
(70)
(98)
(57)
(94)
(58)
(38)
(42)
(46)
(116)
(80)
(99)
(77)
(45)
(67)
Data Points