```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn
```

Data Loading

```python
df = pd.read_csv('loan_sanction_test.csv')
```

Display the Dataset

```python
print(df.head())
```

```
    Loan_ID Gender Married Dependents     Education Self_Employed  \
0  LP001015   Male     Yes          0      Graduate            No   
1  LP001022   Male     Yes          1      Graduate            No   
2  LP001031   Male     Yes          2      Graduate            No   
3  LP001035   Male     Yes          2      Graduate            No   
4  LP001051   Male      No          0  Not Graduate            No   

   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0             5720                  0       110.0             360.0   
1             3076               1500       126.0             360.0   
2             5000               1800       208.0             360.0   
3             2340               2546       100.0             360.0   
4             3276                  0        78.0             360.0   

   Credit_History Property_Area  
0             1.0         Urban  
1             1.0         Urban  
2             1.0         Urban  
3             NaN         Urban  
4             1.0         Urban  
```

Missing Values

```python
# Check for missing values
missing_values = df.isnull().sum()
print(missing_values[missing_values > 0])
```

```
Gender              11
Dependents          10
Self_Employed       23
LoanAmount           5
Loan_Amount_Term     6
Credit_History      29
dtype: int64
```

Identify Numeric and Categorical Columns

```
numeric_cols = df.select_dtypes(include=['number']).columns
categorical_cols = df.select_dtypes(include=['object']).columns

print("Numeric Columns:", numeric_cols)
print("Categorical Columns:", categorical_cols)
```

```
Numeric Columns: Index(['ApplicantIncome', 'CoapplicantIncome',
'LoanAmount',
       'Loan_Amount_Term', 'Credit_History'],
      dtype='object')
Categorical Columns: Index(['Loan_ID', 'Gender', 'Married',
'Dependents', 'Education',
       'Self_Employed', 'Property_Area'],
      dtype='object')
```

Handle Missing Values

```
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].mean())

for col in categorical_cols:
    df[col].fillna(df[col].mode()[0], inplace=True)
```

```
C:\Users\Vashu Jain\AppData\Local\Temp\
ipykernel_14048\3363347938.py:6: FutureWarning: A value is trying to
be set on a copy of a DataFrame or Series through chained assignment
using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  df[col].fillna(df[col].mode()[0], inplace=True)
```

Verify Missing Values

```
missing_values_after = df.isnull().sum()
print(missing_values_after[missing_values_after > 0])
```

```
Series([], dtype: int64)
```

Summarize Basic Statistics

```
summary_stats = df.describe()
print(summary_stats)
```

```
       ApplicantIncome  CoapplicantIncome  LoanAmount
Loan_Amount_Term  \
count       367.000000         367.000000  367.000000
367.000000
mean       4805.599455        1569.577657  136.132597
342.537396
std        4910.685399        2334.232099   60.946040
64.620366
min           0.000000           0.000000   28.000000
6.000000
25%        2864.000000           0.000000  101.000000
360.000000
50%        3786.000000        1025.000000  126.000000
360.000000
75%        5060.000000        2430.500000  157.500000
360.000000
max       72529.000000       24000.000000  550.000000
480.000000

       Credit_History
count      367.000000
mean         0.825444
std          0.364778
min          0.000000
25%          1.000000
50%          1.000000
75%          1.000000
max          1.000000
```

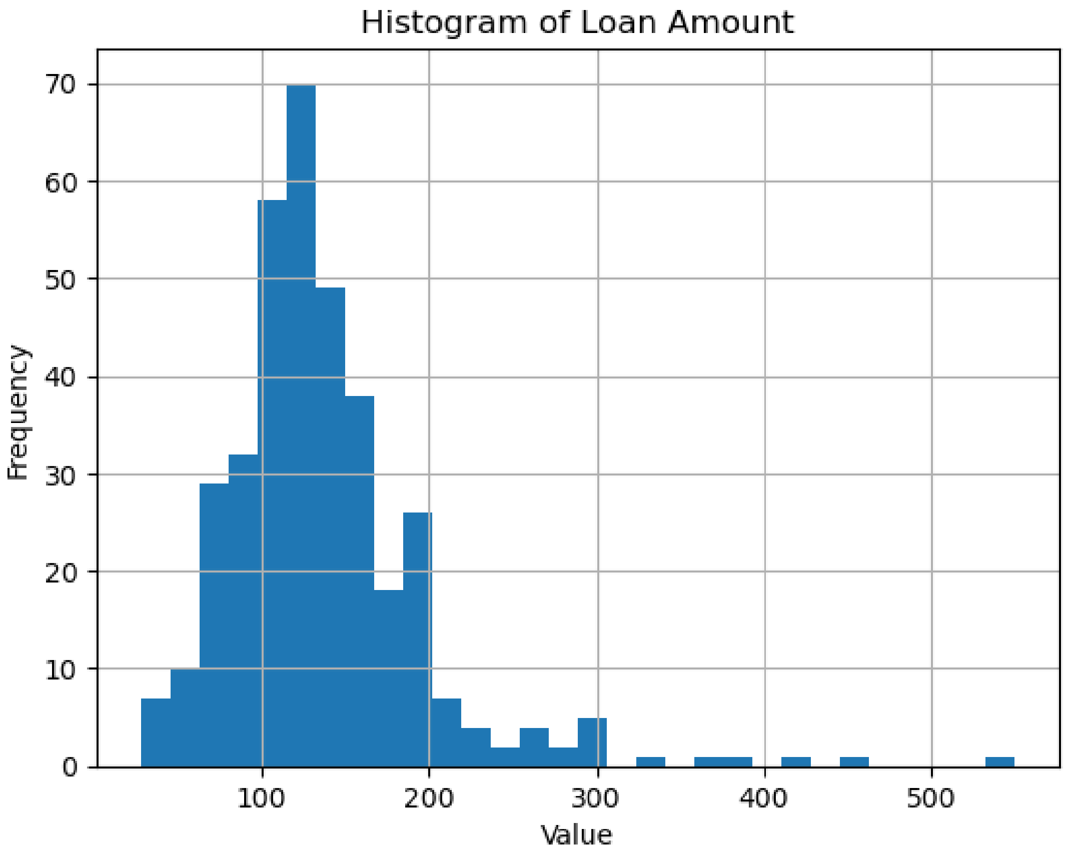
Univariate Analysis

Numeric Variables

```
print(df.columns)

Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome',
'LoanAmount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area'],
      dtype='object')
```

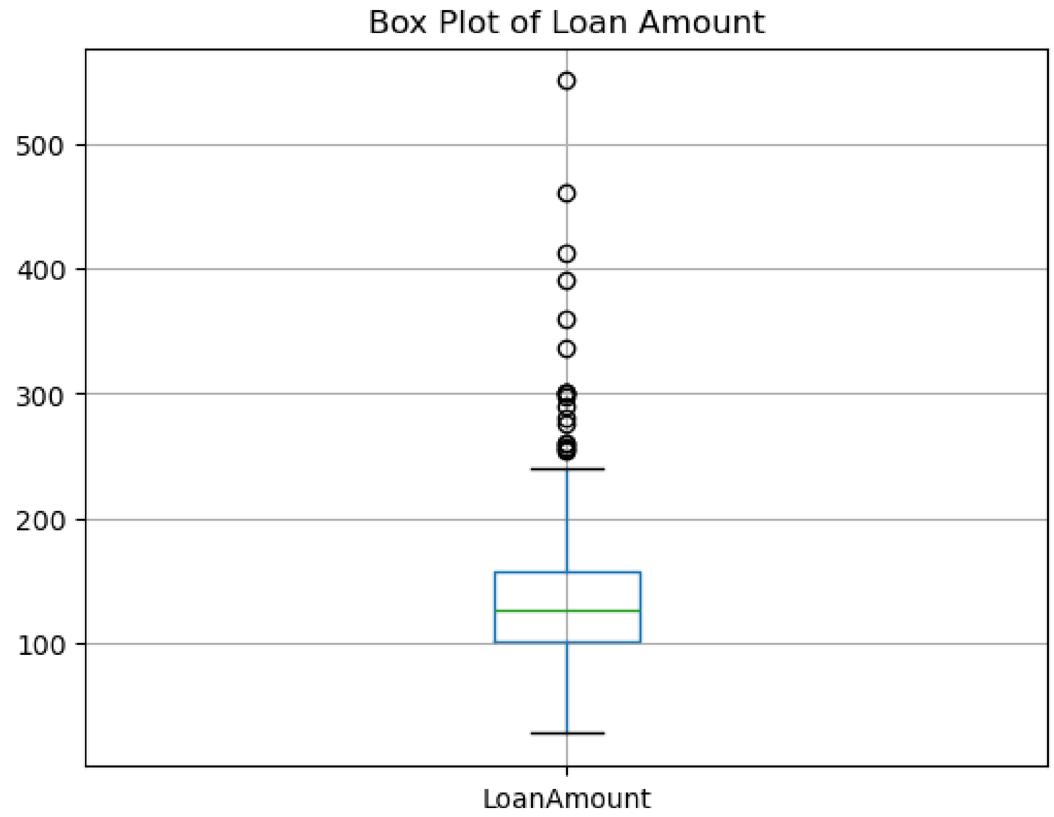
Histograms

```
df['LoanAmount'].hist(bins=30)
plt.title('Histogram of Loan Amount')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.show()
```



Box plot

```
df.boxplot(column='LoanAmount')
plt.title('Box Plot of Loan Amount')
plt.show()
```



column name check

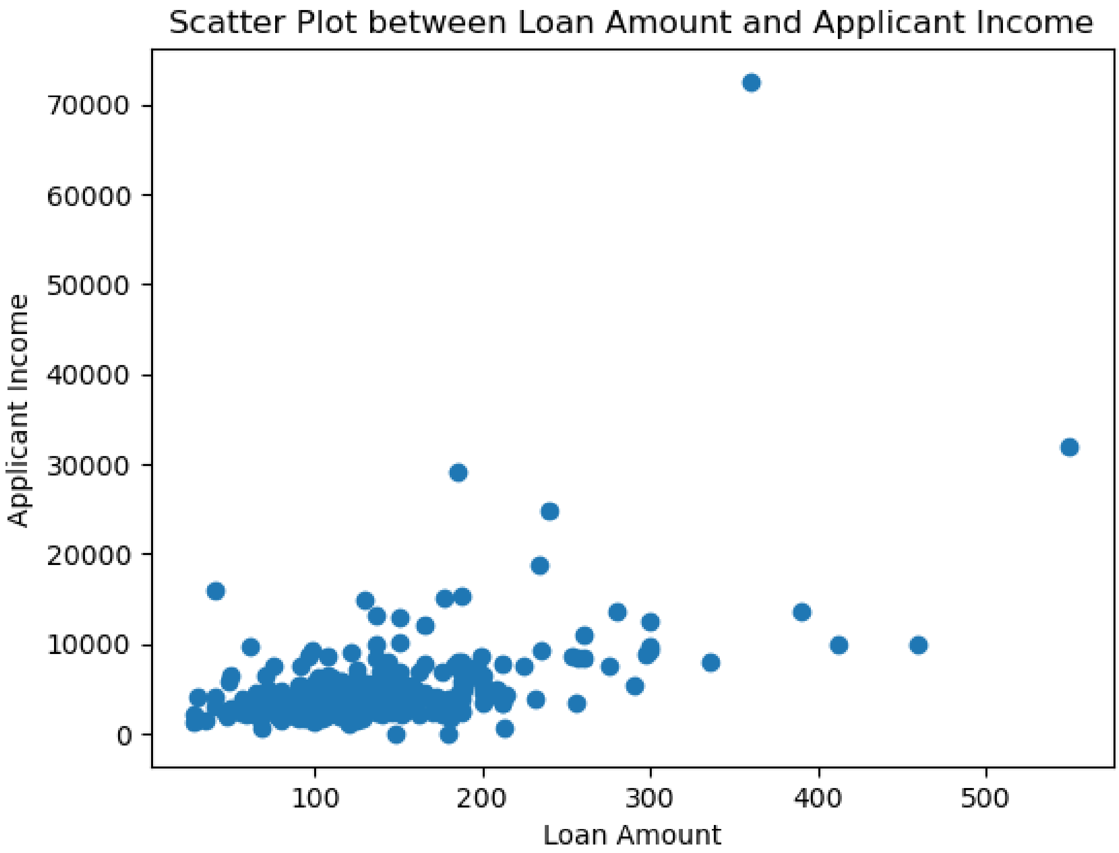
```
print(df.columns.tolist())
```

```
['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
'Loan_Amount_Term', 'Credit_History', 'Property_Area']
```

Bivariate Analysis

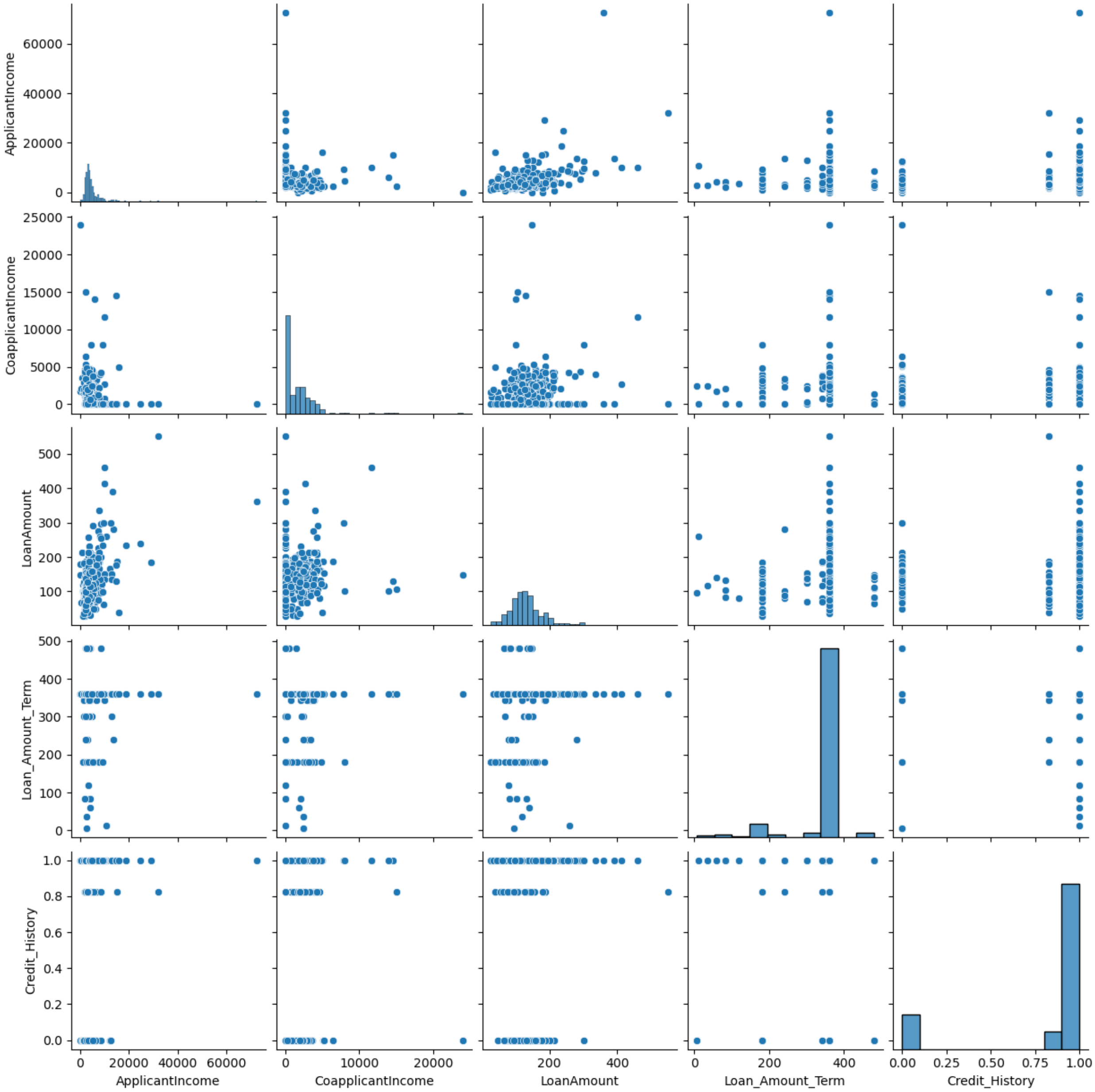
Scatter plot

```
plt.scatter(df['LoanAmount'], df['ApplicantIncome'])
plt.title('Scatter Plot between Loan Amount and Applicant Income')
plt.xlabel('Loan Amount')
plt.ylabel('Applicant Income')
plt.show()
```



Pair plot

```
sns.pairplot(df)
plt.show()
```



Multivariate AnalysiS

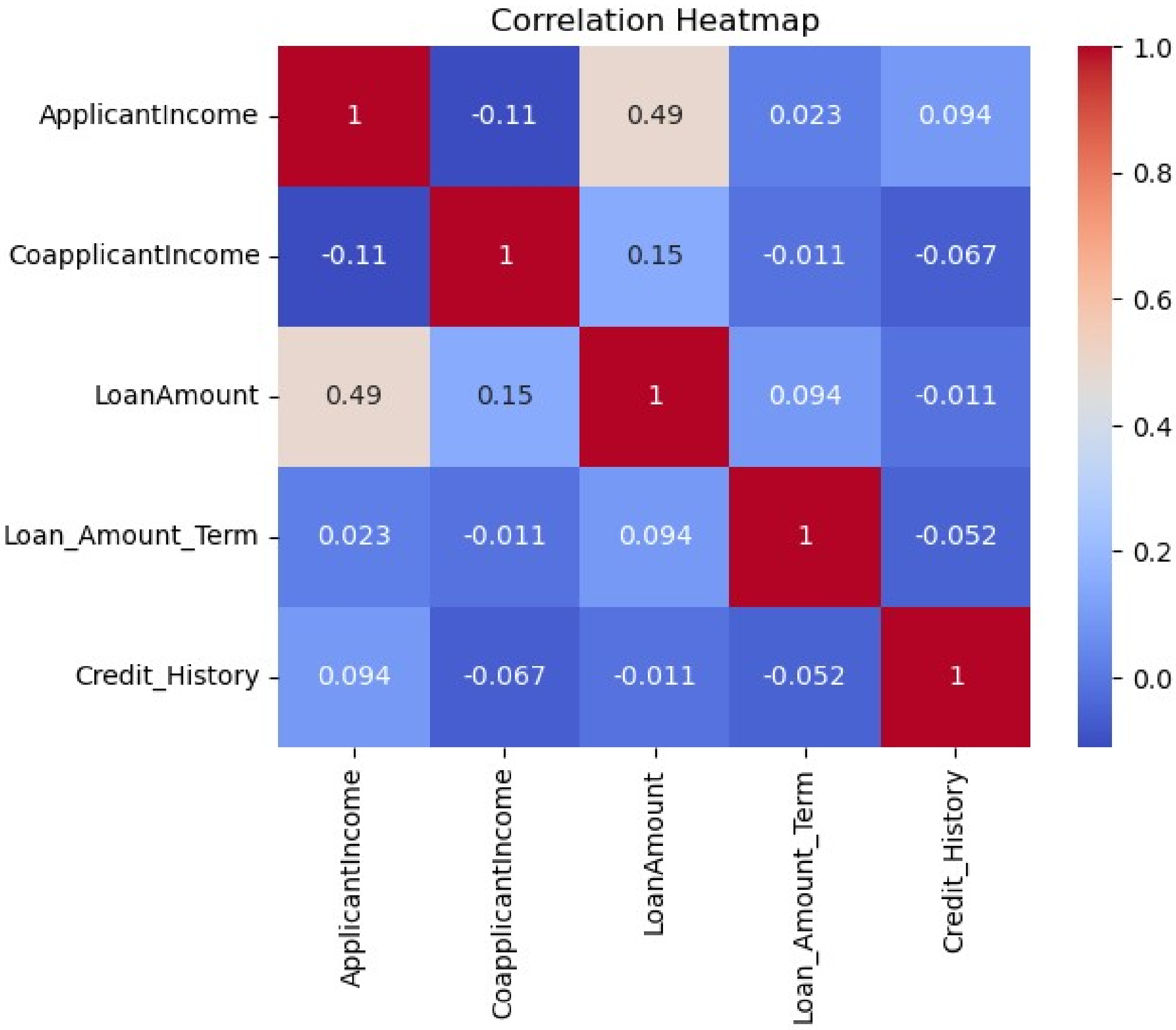
```
numeric_df = df.select_dtypes(include=['number'])

correlation_matrix = numeric_df.corr()
print(correlation_matrix)
```

```
                   ApplicantIncome  CoapplicantIncome  LoanAmount  \
ApplicantIncome           1.000000          -0.110335    0.490174
CoapplicantIncome        -0.110335           1.000000    0.150112
LoanAmount                0.490174           0.150112    1.000000
Loan_Amount_Term          0.023187          -0.010940    0.093856
Credit_History            0.094083          -0.066798   -0.011405
```

```
                   Loan_Amount_Term  Credit_History
ApplicantIncome            0.023187        0.094083
CoapplicantIncome         -0.010940       -0.066798
LoanAmount                 0.093856       -0.011405
Loan_Amount_Term           1.000000       -0.052370
Credit_History            -0.052370        1.000000
```

Heat map

```
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```


Correlation Heatmap
ApplicantIncome
CoapplicantIncome
LoanAmount
Loan_Amount_Term
Credit_History
ApplicantIncome
CoapplicantIncome
LoanAmount
Loan_Amount_Term
Credit_History