Data Collection and Preprocessing Phase

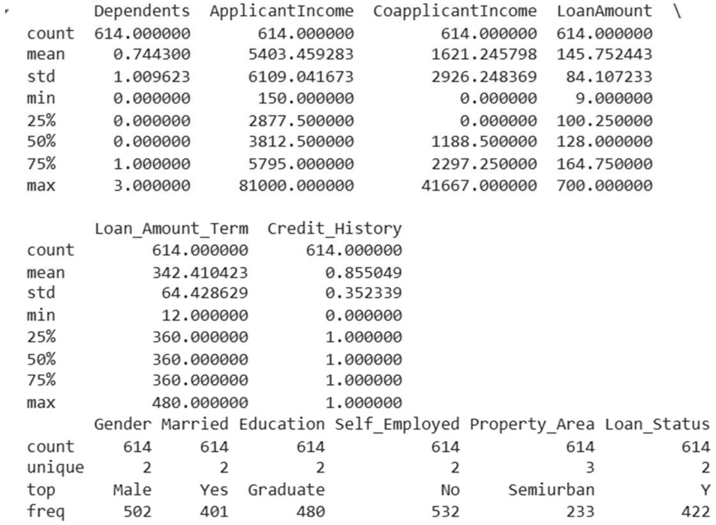
Date	20 June 2025
Team ID	SWTID1749791625
Project Title	Smart Lender- Applicant Credibility Prediction for Loan Approval
Maximum Marks	6 Marks

Data Exploration and Preprocessing Report

Dataset variables will be statistically analyzed to identify patterns and outliers. Python will be used for preprocessing tasks such as normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring high-quality input for analysis and modeling, and providing a strong foundation for reliable insights and predictions.

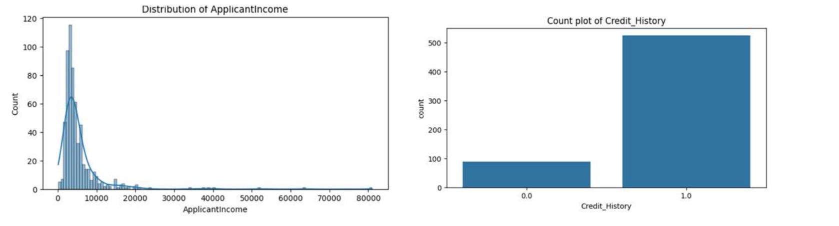
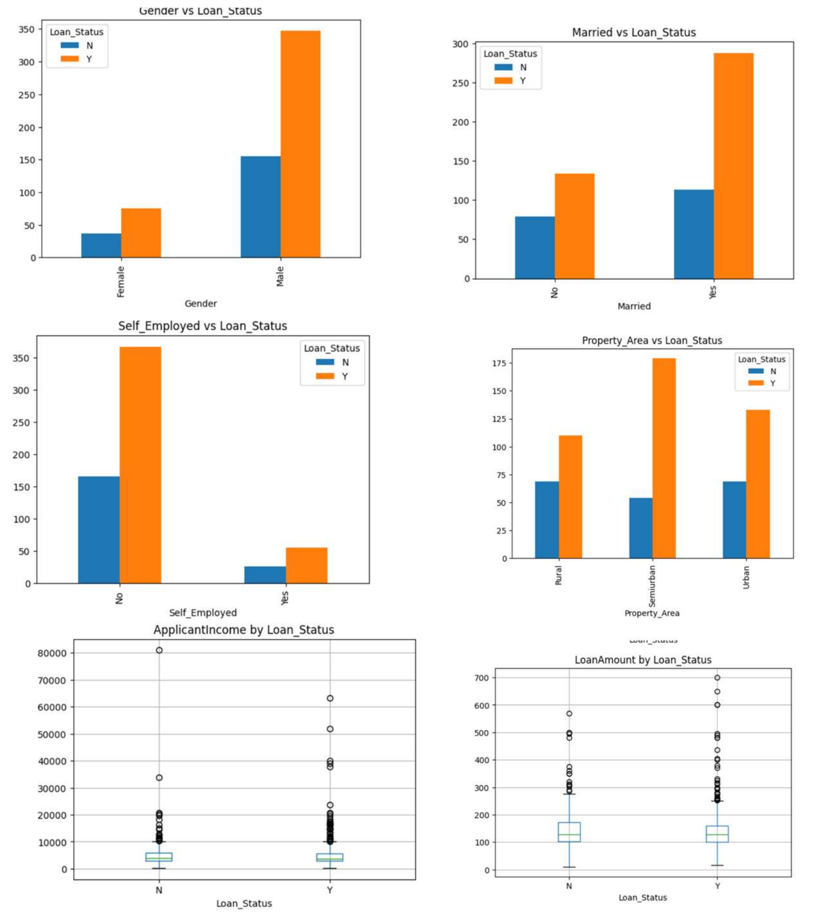
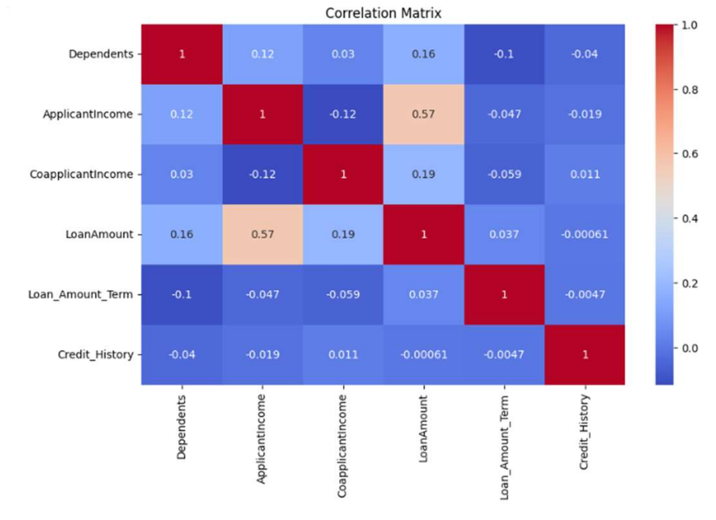
<table>
<tr><th>Section</th><th>Description</th></tr>
<tr><td>Data Overview</td><td>

Dimension:
614rows×13columns.
Descriptive Statistics:

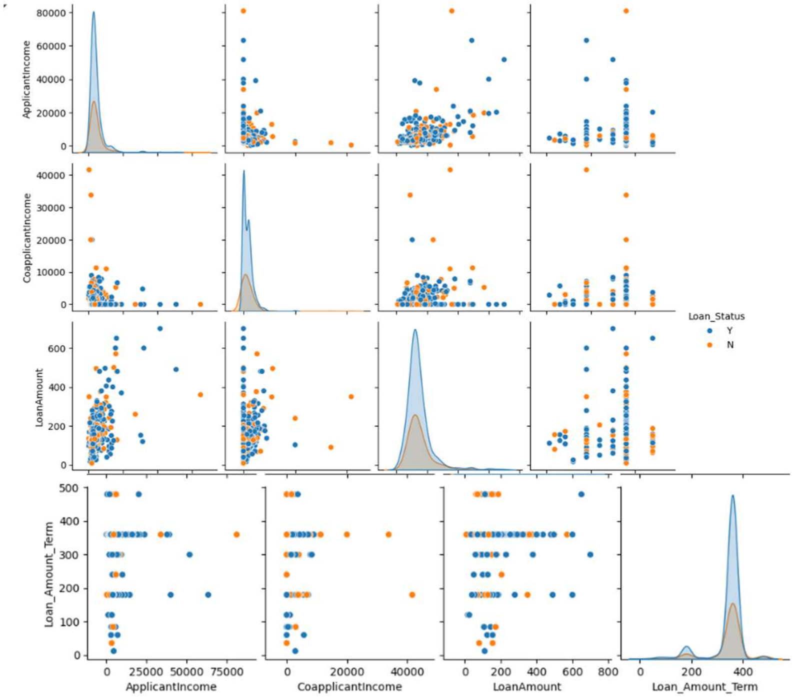


```
       Dependents  ApplicantIncome  CoapplicantIncome  LoanAmount  \
count  614.000000       614.000000         614.000000  614.000000
mean     0.744300      5403.459283        1621.245798  145.752443
std      1.009623      6109.041673        2926.248369   84.107233
min      0.000000       150.000000           0.000000    9.000000
25%      0.000000      2877.500000           0.000000  100.250000
50%      0.000000      3812.500000        1188.500000  128.000000
75%      1.000000      5795.000000        2297.250000  164.750000
max      3.000000     81000.000000       41667.000000  700.000000

       Loan_Amount_Term  Credit_History
count        614.000000      614.000000
mean         342.410423        0.855049
std           64.428629        0.352339
min           12.000000        0.000000
25%          360.000000        1.000000
50%          360.000000        1.000000
75%          360.000000        1.000000
max          480.000000        1.000000
       Gender Married Education Self_Employed Property_Area Loan_Status
count     614     614       614           614           614         614
unique      2       2         2             2             3           2
top      Male     Yes  Graduate            No     Semiurban           Y
freq      502     401       480           532           233         422
```

</td></tr>
</table>

Univariate Analysis	
Bivariate Analysis	
Multivariate Analysis	

	Dependents	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
Dependents	1	0.12	0.03	0.16	-0.1	-0.04
ApplicantIncome	0.12	1	-0.12	0.57	-0.047	-0.019
CoapplicantIncome	0.03	-0.12	1	0.19	-0.059	0.011
LoanAmount	0.16	0.57	0.19	1	0.037	-0.00061
Loan_Amount_Term	-0.1	-0.047	-0.059	0.037	1	-0.0047
Credit_History	-0.04	-0.019	0.011	-0.00061	-0.0047	1

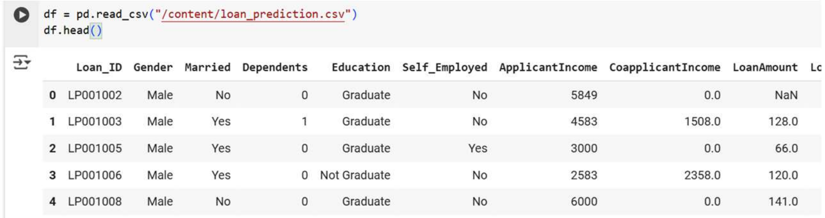


Outliers and Anomalies

-

Data Preprocessing Code Screenshots

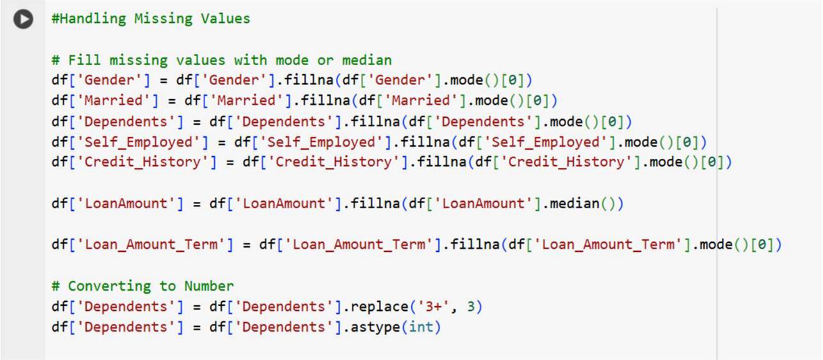
Loading Data



```
df = pd.read_csv("/content/loan_prediction.csv")
df.head()
```

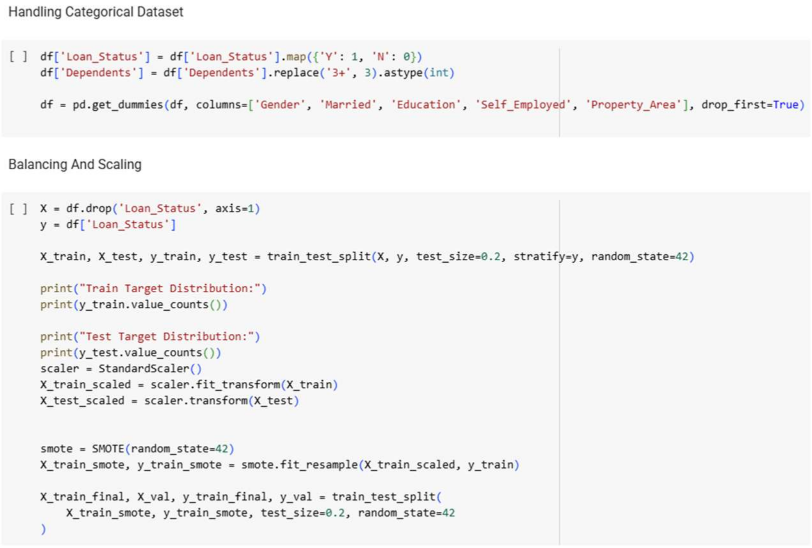
```
   Loan_ID  Gender Married Dependents     Education Self_Employed ApplicantIncome CoapplicantIncome LoanAmount Lo
0  LP001002   Male      No          0      Graduate            No            5849               0.0        NaN
1  LP001003   Male     Yes          1      Graduate            No            4583            1508.0      128.0
2  LP001005   Male     Yes          0      Graduate           Yes            3000               0.0       66.0
3  LP001006   Male     Yes          0  Not Graduate            No            2583            2358.0      120.0
4  LP001008   Male      No          0      Graduate            No            6000               0.0      141.0
```

Handling Missing Data



```
#Handling Missing Values

# Fill missing values with mode or median
df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
df['Married'] = df['Married'].fillna(df['Married'].mode()[0])
df['Dependents'] = df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed'] = df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mode()[0])

df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].median())

df['Loan_Amount_Term'] = df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0])

# Converting to Number
df['Dependents'] = df['Dependents'].replace('3+', 3)
df['Dependents'] = df['Dependents'].astype(int)
```

<table>
<tr>
<td>Data Transformation</td>
<td>



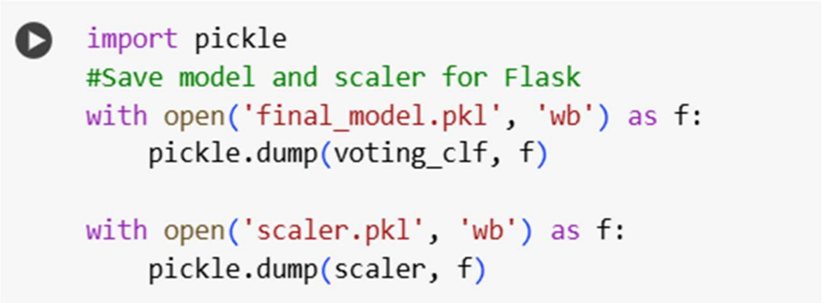
Handling Categorical Dataset

```
[ ] df['Loan_Status'] = df['Loan_Status'].map({'Y': 1, 'N': 0})
    df['Dependents'] = df['Dependents'].replace('3+', 3).astype(int)

    df = pd.get_dummies(df, columns=['Gender', 'Married', 'Education', 'Self_Employed', 'Property_Area'], drop_first=True)
```

Balancing And Scaling

```
[ ] X = df.drop('Loan_Status', axis=1)
    y = df['Loan_Status']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)

    print("Train Target Distribution:")
    print(y_train.value_counts())

    print("Test Target Distribution:")
    print(y_test.value_counts())
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)


    smote = SMOTE(random_state=42)
    X_train_smote, y_train_smote = smote.fit_resample(X_train_scaled, y_train)

    X_train_final, X_val, y_train_final, y_val = train_test_split(
        X_train_smote, y_train_smote, test_size=0.2, random_state=42
    )
```

</td>
</tr>
<tr>
<td>Feature Engineering</td>
<td>Attached the codes in final submission.</td>
</tr>
<tr>
<td>Save Processed Data</td>
<td>



```
import pickle
#Save model and scaler for Flask
with open('final_model.pkl', 'wb') as f:
    pickle.dump(voting_clf, f)

with open('scaler.pkl', 'wb') as f:
    pickle.dump(scaler, f)
```

</td>
</tr>
</table>