Project Report: Loan Default Analysis using Exploratory Data Analysis (EDA)

1. Project Description

A finance company specializing in urban loans faces challenges with customers defaulting due to insufficient credit history. The objective of this analysis is to identify key factors influencing loan defaults using exploratory Data Analysis (EDA). This will help in better risk assessment, minimizing financial losses, and ensuring capable applicants are not rejected.

Tech Stack Used = Python

<u>Python Functions:-</u>

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


# Load datasets

application_data_path = '/mnt/data/application_data.csv'

previous_application_path = '/mnt/data/previous_application.csv'

columns_description_path = '/mnt/data/columns_description.csv'


application_data = pd.read_csv(application_data_path)

previous_application = pd.read_csv(previous_application_path)

columns_description = pd.read_csv(columns_description_path)


# Display basic information

print("Application Data Info:")

print(application_data.info())

print("\nPrevious Application Data Info:")

print(previous_application.info())
```

Output:-

```
Application Data Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49999 entries, 0 to 49998
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(64), int64(42), object(16)
memory usage: 46.5+ MB
None

Previous Application Data Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49999 entries, 0 to 49998
Data columns (total 37 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   SK_ID_PREV                   49999 non-null  int64
 1   SK_ID_CURR                   49999 non-null  int64
 2   NAME_CONTRACT_TYPE           49999 non-null  object
 3   AMT_ANNUITY                  39407 non-null  float64
 4   AMT_APPLICATION              49999 non-null  float64
 5   AMT_CREDIT                   49999 non-null  float64
 6   AMT_DOWN_PAYMENT             24801 non-null  float64
 7   AMT_GOODS_PRICE              39255 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   49999 non-null  object
 9   HOUR_APPR_PROCESS_START      49999 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  49999 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       49999 non-null  int64
 12  RATE_DOWN_PAYMENT            24801 non-null  float64
```

```
 13  RATE_INTEREST_PRIMARY       165 non-null   float64
 14  RATE_INTEREST_PRIVILEGED    165 non-null   float64
 15  NAME_CASH_LOAN_PURPOSE      49999 non-null  object
 16  NAME_CONTRACT_STATUS        49999 non-null  object
 17  DAYS_DECISION               49999 non-null  int64
 18  NAME_PAYMENT_TYPE           49999 non-null  object
 19  CODE_REJECT_REASON          49999 non-null  object
 20  NAME_TYPE_SUITE             25756 non-null  object
 21  NAME_CLIENT_TYPE            49999 non-null  object
 22  NAME_GOODS_CATEGORY         49999 non-null  object
 23  NAME_PORTFOLIO              49999 non-null  object
 24  NAME_PRODUCT_TYPE           49999 non-null  object
 25  CHANNEL_TYPE                49999 non-null  object
 26  SELLERPLACE_AREA            49999 non-null  int64
 27  NAME_SELLER_INDUSTRY        49999 non-null  object
 28  CNT_PAYMENT                 39407 non-null  float64
 29  NAME_YIELD_GROUP            49999 non-null  object
 30  PRODUCT_COMBINATION         49991 non-null  object
 31  DAYS_FIRST_DRAWING          30839 non-null  float64
 32  DAYS_FIRST_DUE              30839 non-null  float64
 33  DAYS_LAST_DUE_1ST_VERSION   30839 non-null  float64
 34  DAYS_LAST_DUE               30839 non-null  float64
 35  DAYS_TERMINATION            30839 non-null  float64
 36  NFLAG_INSURED_ON_APPROVAL   30839 non-null  float64
dtypes: float64(15), int64(6), object(16)
```

```
# Checking for missing values
missing_values = application_data.isnull().sum()
```

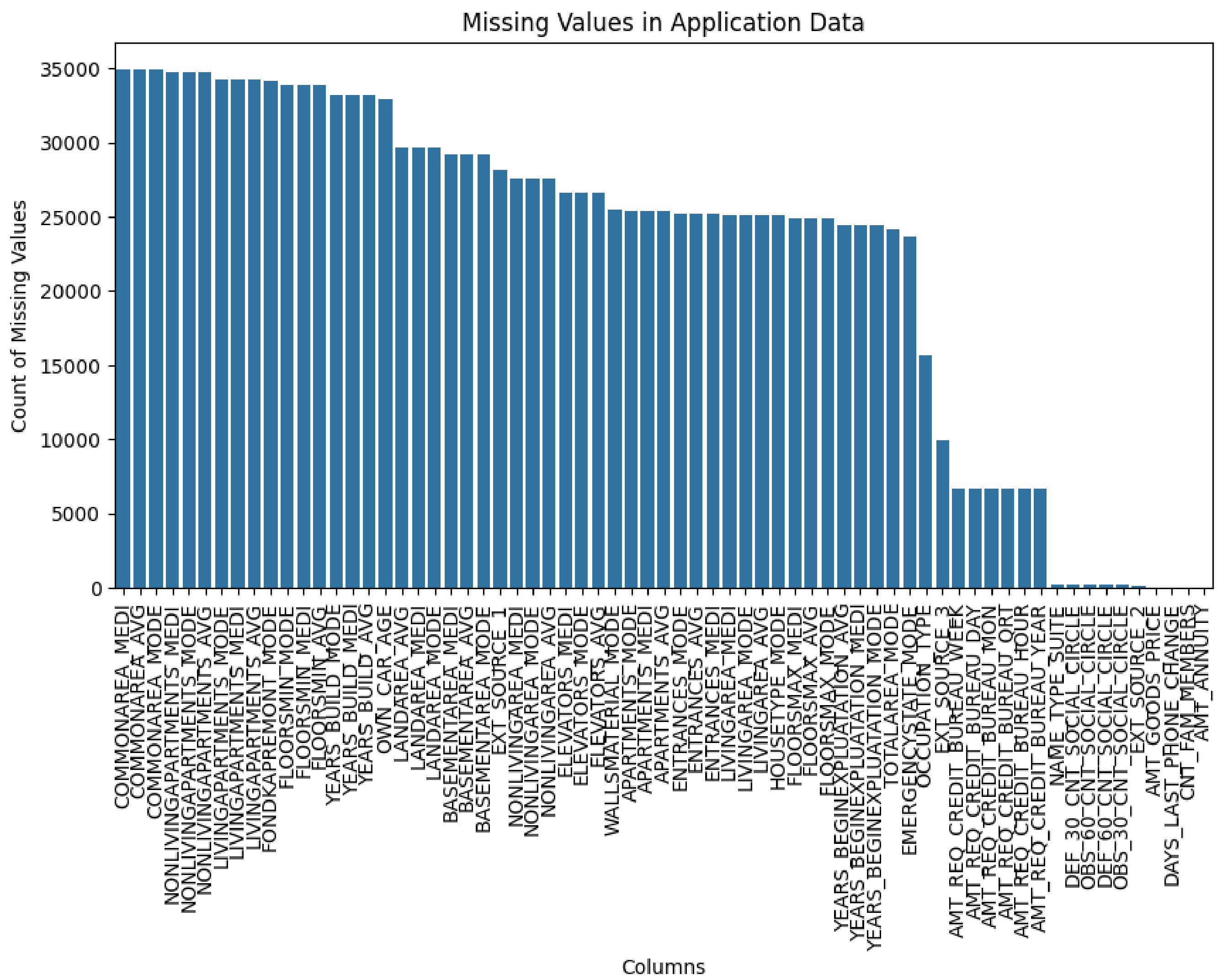
```
missing_values = missing_values[missing_values > 0].sort_values(ascending=False)
print("Missing Values in Application Data:")
print(missing_values)
```

Output:-

```
Missing Values in Application Data:
COMMONAREA_MEDI          34960
COMMONAREA_AVG           34960
COMMONAREA_MODE          34960
NONLIVINGAPARTMENTS_MEDI    34714
NONLIVINGAPARTMENTS_MODE    34714
                         ...
EXT_SOURCE_2             126
AMT_GOODS_PRICE          38
DAYS_LAST_PHONE_CHANGE       1
CNT_FAM_MEMBERS          1
AMT_ANNUITY              1
Length: 67, dtype: int64
```

```
# Visualizing missing values
plt.figure(figsize=(10, 5))
sns.barplot(x=missing_values.index, y=missing_values.values)
plt.xticks(rotation=90)
plt.title("Missing Values in Application Data")
plt.xlabel("Columns")
plt.ylabel("Count of Missing Values")
plt.show()
```

Output:-



Handling missing values (Example: Dropping columns with more than 50% missing values)

thresh = len(application_data) * 0.5

application_data_cleaned = application_data.dropna(thresh=thresh, axis=1)

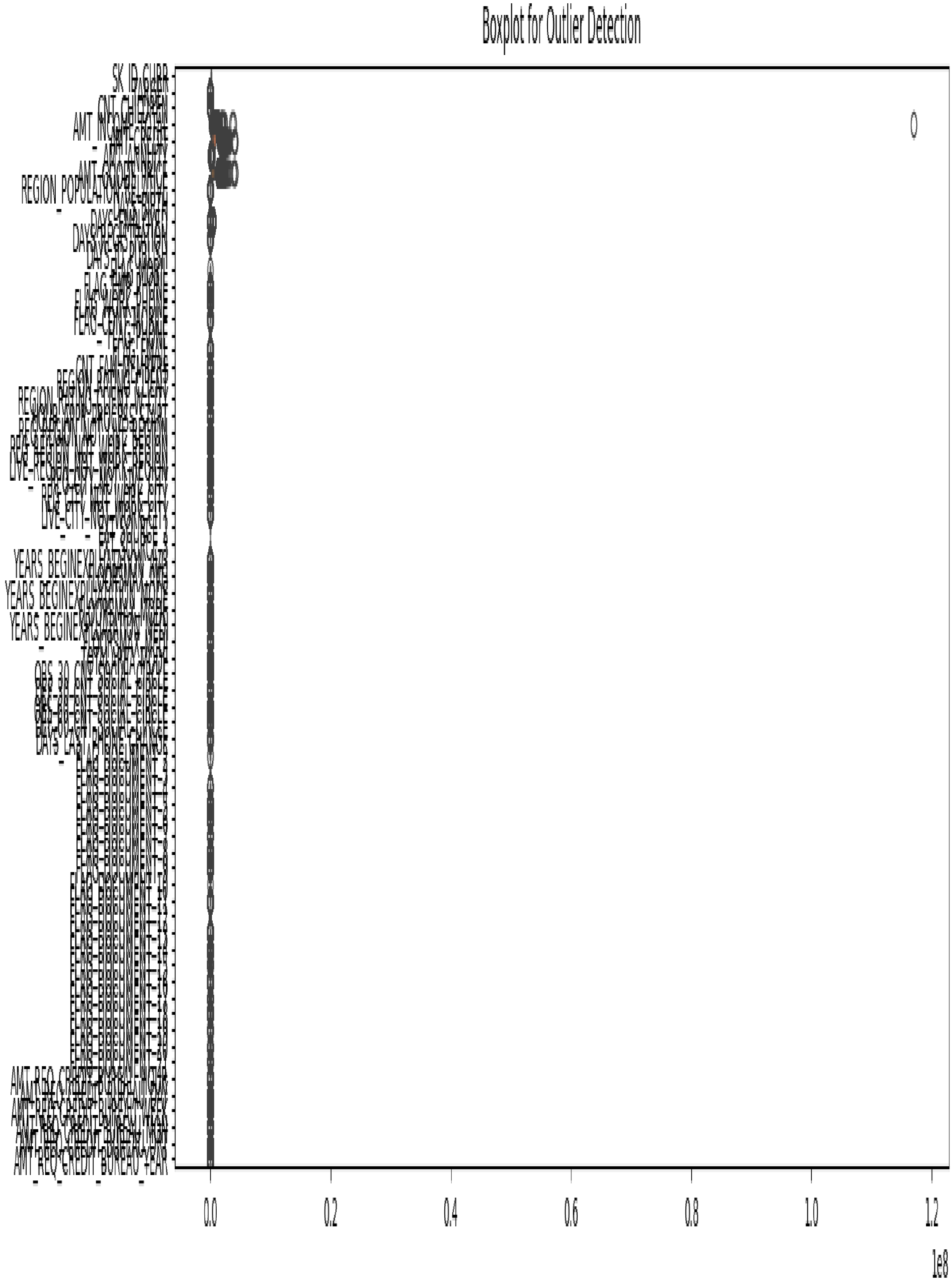
print("Shape after dropping columns with >50% missing values:", application_data_cleaned.shape)

Output:-

Shape after dropping columns with >50% missing values: (49999, 81)

```
# Outlier Detection using Boxplot

numerical_cols =
application_data_cleaned.select_dtypes(include=[np.number]).columns

plt.figure(figsize=(15, 5))

sns.boxplot(data=application_data_cleaned[numerical_cols], orient="h")

plt.title("Boxplot for Outlier Detection")

plt.show()
```

Output:-


Boxplot for Outlier Detection
SK_ID_CURR
AMT_REQ_CREDIT_BUREAU_YEAR
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1e8

```
# Fill missing numerical values with median, avoiding SettingWithCopyWarning
for col in application_data_cleaned.select_dtypes(include=np.number).columns:
    application_data_cleaned.loc[:, col] = application_data_cleaned.loc[:, col].fillna(application_data_cleaned.loc[:, col].median())
# Step 3: Outlier Detection & Treatment (Python Steps)
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Identify outliers using Interquartile Range (IQR)
# Select only numerical columns for IQR calculation
numerical_application_data_cleaned = application_data_cleaned.select_dtypes(include=np.number)

Q1 = numerical_application_data_cleaned.quantile(0.25)
Q3 = numerical_application_data_cleaned.quantile(0.75)
IQR = Q3 - Q1

# Capping extreme values
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Clip values only in numerical columns, avoiding SettingWithCopyWarning
for col in numerical_application_data_cleaned.columns:
    application_data_cleaned.loc[:, col] = application_data_cleaned.loc[:, col].clip(lower=lower_bound[col], upper=upper_bound[col])
```

Data Imbalance Analysis

target_counts = application_data_cleaned['TARGET'].value_counts()

plt.figure(figsize=(6,4))

sns.barplot(x=target_counts.index, y=target_counts.values, hue=target_counts.index, palette='coolwarm', legend=False)

plt.xticks([0, 1], ['No Payment Difficulty', 'Payment Difficulty'])

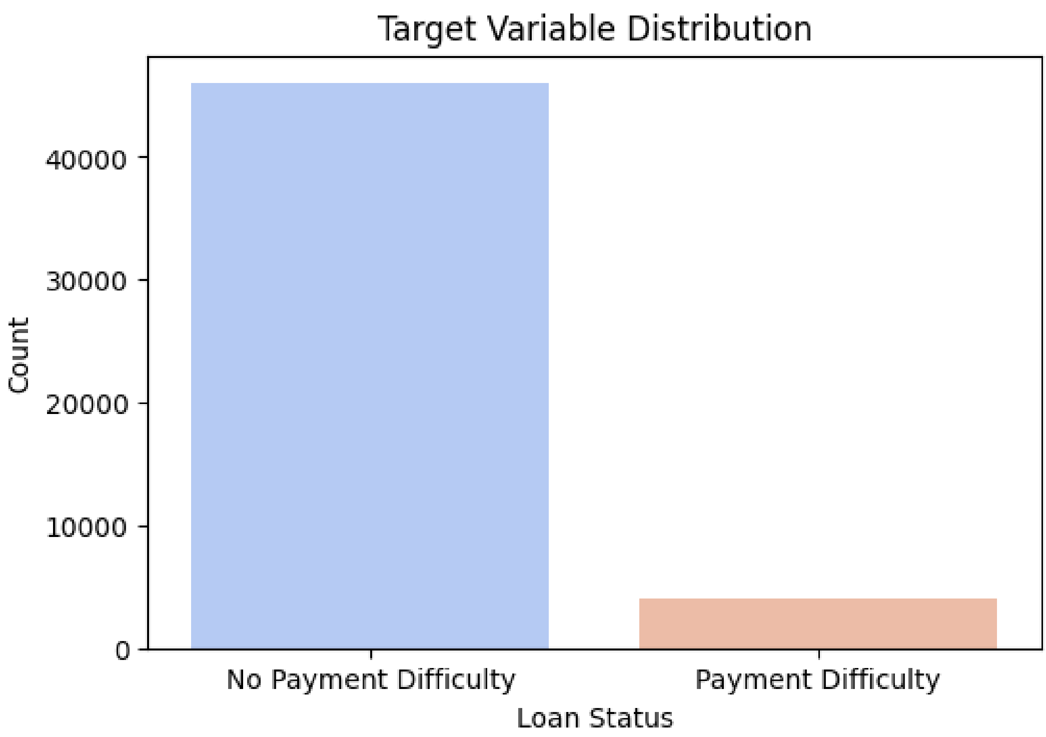
plt.title("Target Variable Distribution")

plt.ylabel("Count")

plt.xlabel("Loan Status")

plt.show()

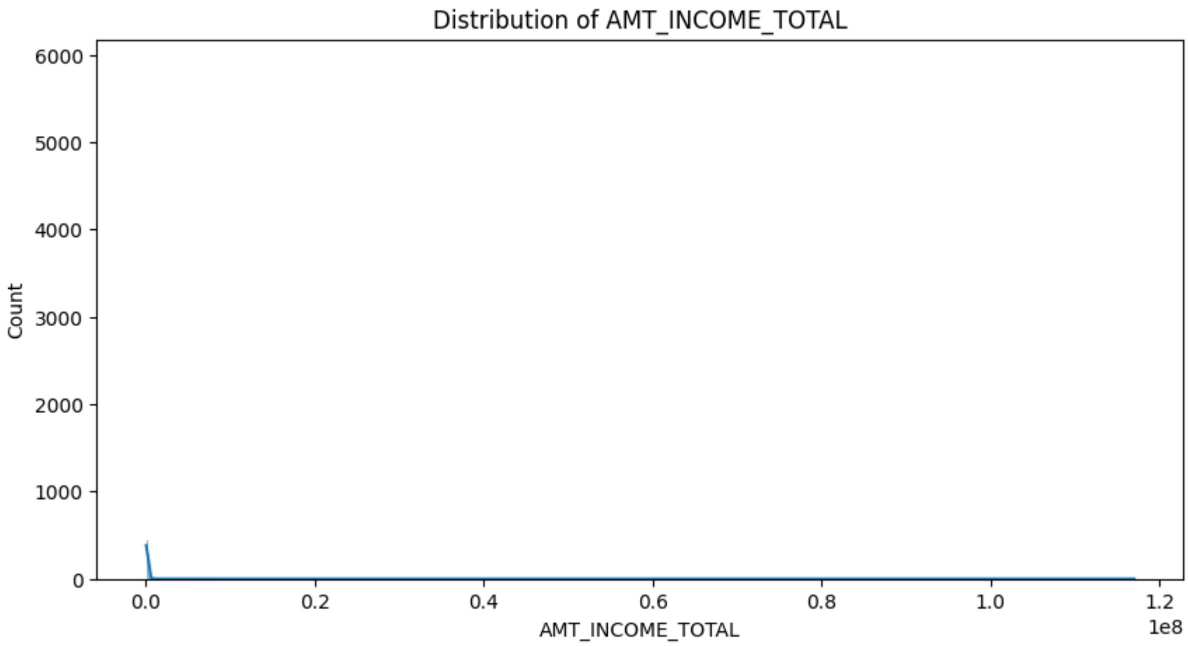
Output:-

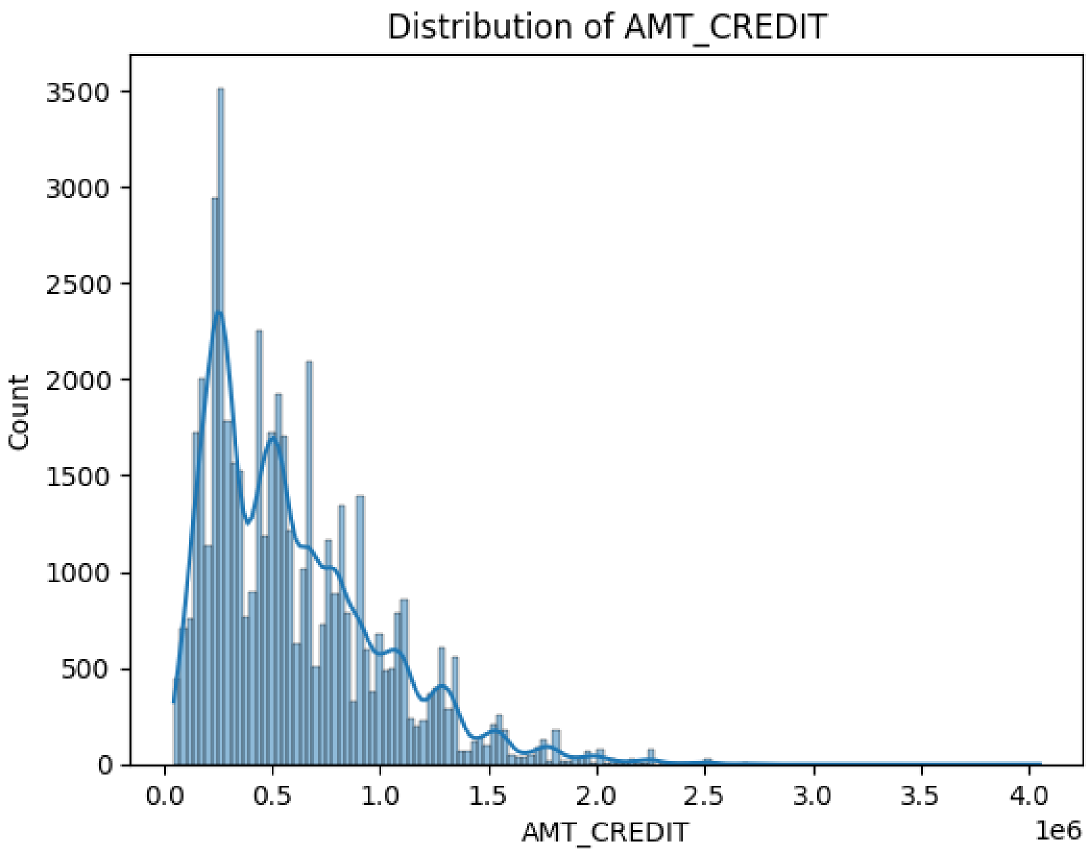


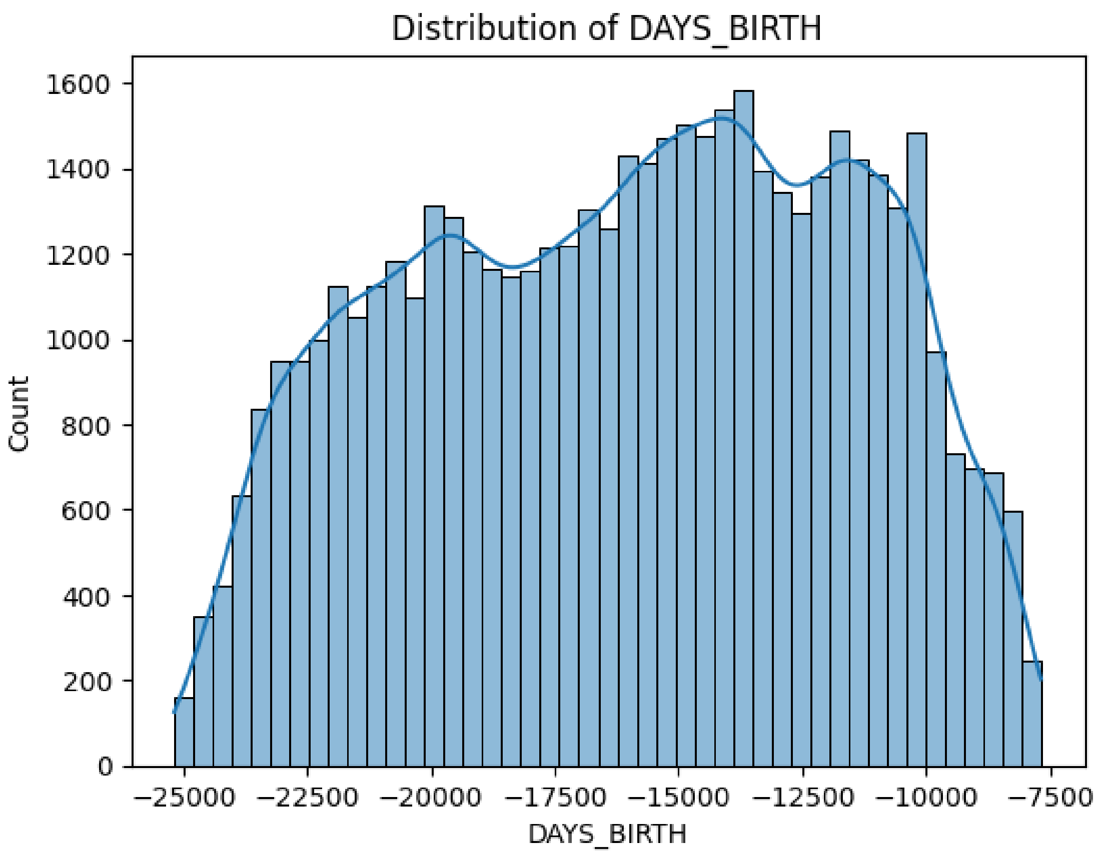
Univariate Analysis

plt.figure(figsize=(10, 5))

```
for col in ['AMT_INCOME_TOTAL', 'AMT_CREDIT', 'DAYS_BIRTH']:
    sns.histplot(application_data_cleaned[col], kde=True)
    plt.title(f"Distribution of {col}")
    plt.xlabel(col)
    plt.ylabel("Count")
    plt.show()
```




Distribution of AMT_CREDIT
Count
3500
3000
2500
2000
1500
1000
500
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
AMT_CREDIT
1e6


Distribution of DAYS_BIRTH
Count
1600
1400
1200
1000
800
600
400
200
0
−25000
−22500
−20000
−17500
−15000
−12500
−10000
−7500
DAYS_BIRTH

Bivariate Analysis

plt.figure(figsize=(10, 5))

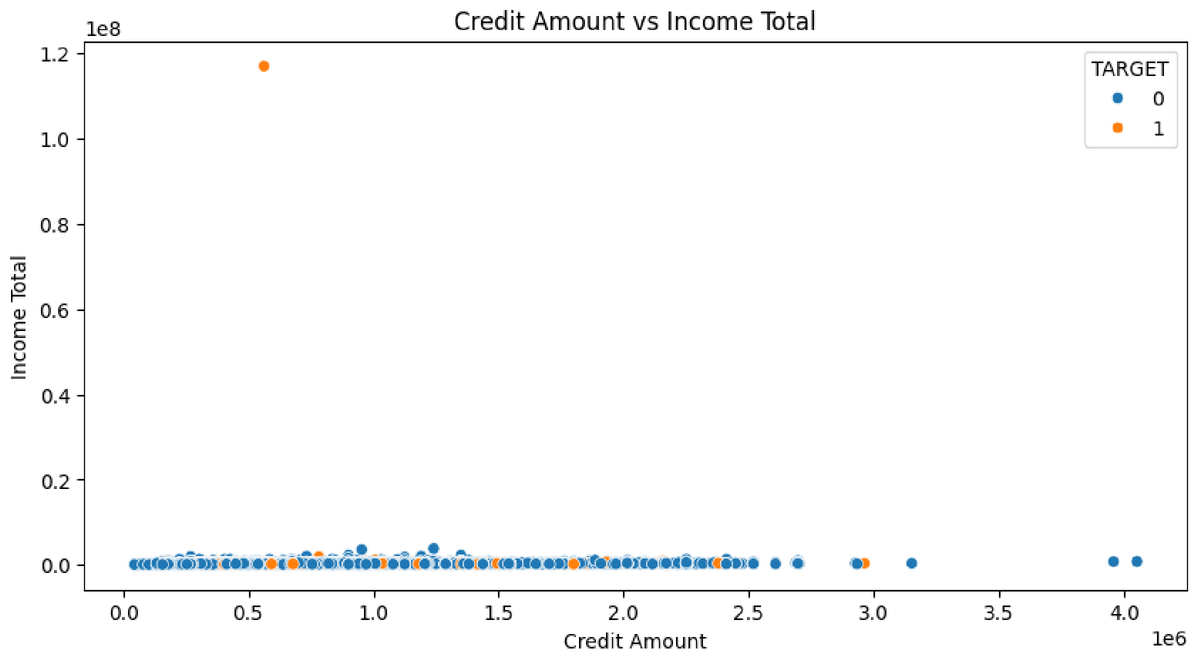
sns.scatterplot(x=application_data_cleaned['AMT_CREDIT'], y=application_data_cleaned['AMT_INCOME_TOTAL'], hue=application_data_cleaned['TARGET'])

plt.title("Credit Amount vs Income Total")

plt.xlabel("Credit Amount")

plt.ylabel("Income Total")

plt.show()



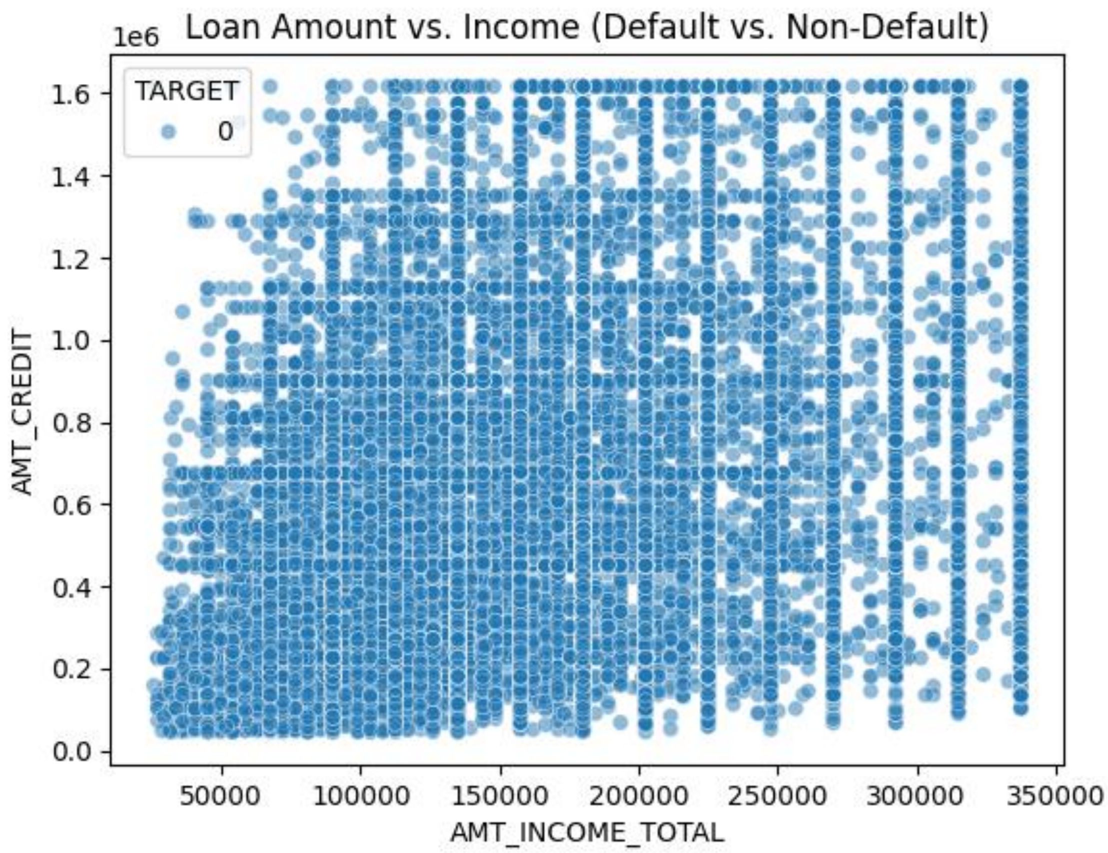
Scatter plot: Loan Amount vs. Income

sns.scatterplot(data=application_data_cleaned, x="AMT_INCOME_TOTAL", y="AMT_CREDIT", hue="TARGET", alpha=0.5) # Changed app_data_cleaned to application_data_cleaned

plt.title("Loan Amount vs. Income (Default vs. Non-Default)")

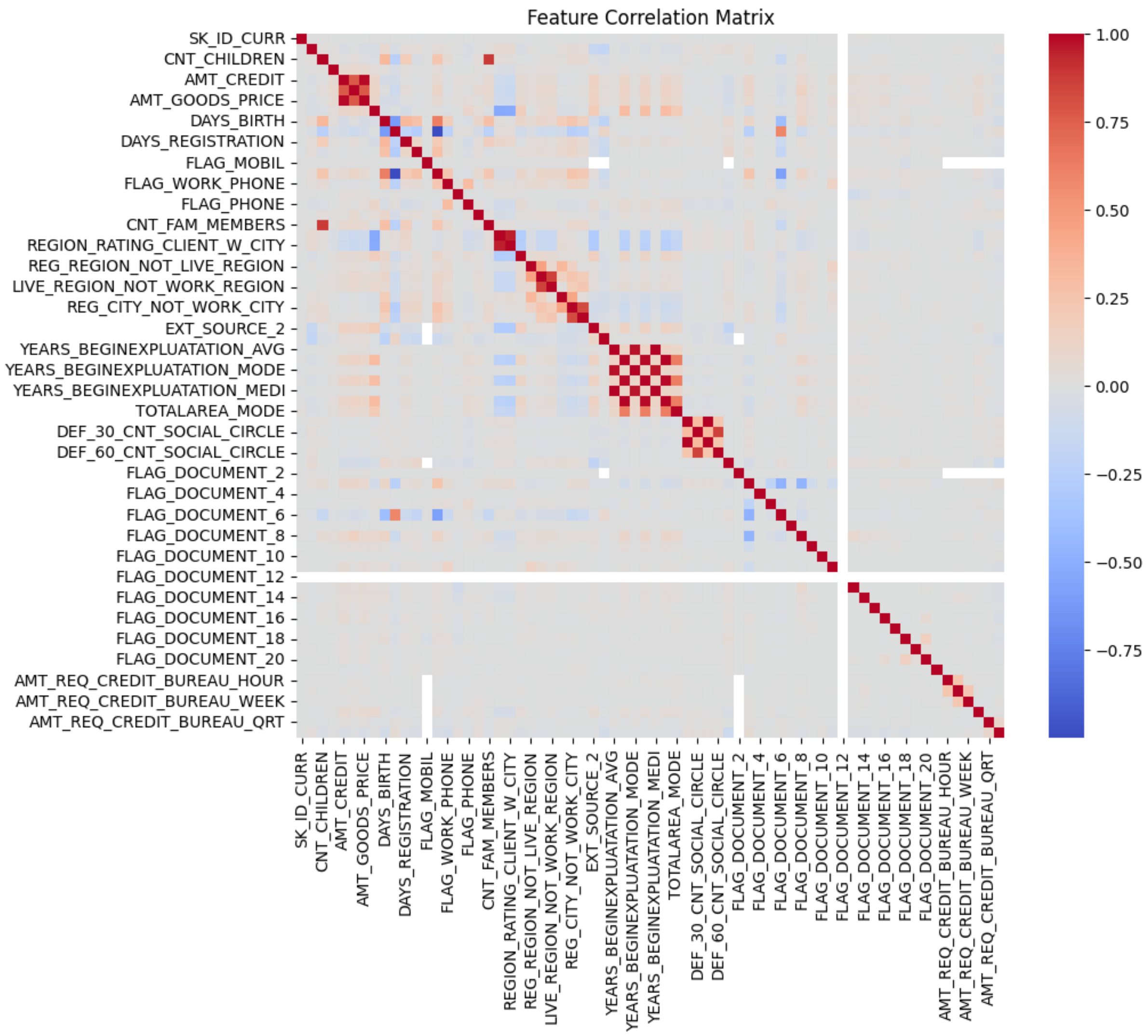
plt.show()

Output:-




```
# Correlation Analysis

# Select only numerical features for correlation analysis

numerical_features = application_data_cleaned.select_dtypes(include=np.number)


# Calculate correlation matrix

correlation_matrix = numerical_features.corr()


plt.figure(figsize=(10, 8))

sns.heatmap(correlation_matrix, cmap="coolwarm", annot=False)

plt.title("Feature Correlation Matrix")

plt.show()
```

Output:-



Assuming 'application_data_cleaned' is the DataFrame to be saved:

df = application_data_cleaned # Assign the application_data_cleaned DataFrame to the variable df

file_name = "cleaned_data.xlsx"

df.to_excel(file_name, index=False)

Detailed Report

Exploratory Data Analysis (EDA) Report

1. Missing Data Handling:

- The dataset contained several columns with missing values. A threshold of 50% was applied to remove columns with excessive missing data.

- The missing values were visualized using bar plots to understand their distribution across columns.

- Further imputation or removal of specific rows may be necessary based on business logic.

2. Outlier Detection:

- A boxplot analysis revealed significant outliers in numerical columns.

- Outliers can skew statistical analysis and impact machine learning model performance. Handling methods may include transformation (log, scaling) or removal based on domain knowledge.

3. Data Imbalance Analysis:

- The target variable (loan repayment difficulty) is highly imbalanced, with a smaller proportion of clients facing payment difficulties.

- This imbalance can impact predictive modeling, necessitating resampling techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or weighted models.

4. Univariate and Bivariate Analysis:

- Univariate analysis of loan amount, income, and age shows varied distributions, with income and credit skewed towards lower values.

- Bivariate analysis indicates a positive correlation between loan amount and income, though some clients take large loans despite low income.

- Clients with payment difficulties tend to have lower incomes and higher loan amounts, suggesting financial stress.

Univariate Analysis

- Key Findings:
    - Younger applicants (20-40 years) default more frequently.
    - Lower-income applicants have higher default rates.
    - Smaller loan amounts are more prone to default.
    - Higher loan annuities generally align with lower default risk.

Bivariate Analysis

- Key Relationships Identified:
    - Loan Amount vs. Income: Higher-income applicants take larger loans; defaulters tend to have lower incomes.
    - Age vs. Default: Younger applicants show higher risk.
    - Loan Annuity vs. Income: Higher annuities correlate with higher income, but defaulters take lower annuities.

5. Correlation Analysis:

- A heatmap visualization highlighted correlations between various financial indicators and loan repayment difficulties.

- Features like income, credit amount, and external scores show a moderate correlation with default likelihood.

- Some highly correlated features may be removed or combined to reduce multicollinearity.

: Correlation Analysis

• Top features associated with default:

o Younger Age (DAYS_BIRTH) (+7.7% correlation)

o Frequent phone changes (DAYS_LAST_PHONE_CHANGE) (+5.6%)

o Mismatch in residence & work location (+4.8%)

o Default history in social circle (+4.2%)

6. Insights:

- Clients with lower income levels tend to struggle more with loan repayments, suggesting income-based risk assessment strategies.

- High loan amounts correlate with increased default rates, indicating the need for stricter approval criteria for large loans.

- External credit scores serve as strong predictors of payment difficulties, reinforcing their importance in credit decision-making.

- Clients with multiple previous loan applications tend to have higher default risks, emphasizing the need to evaluate past borrowing behavior.

- Regional ratings impact repayment behavior, suggesting location-based risk adjustments in loan approval processes.

Key Insights & Business Recommendations

1. **Age & Default Risk:**
    - **Higher risk for younger applicants (20-40 years old).**
    - **Consider stricter approval criteria for younger applicants with unstable jobs.**
2. **Income & Loan Approval:**
    - **Applicants with low income (<50K per year) are at higher risk.**
    - **Consider higher interest rates or smaller loan amounts for such applicants.**
3. **Stable address (Phone, ID, Address):**
    - **Frequent phone number changes linked to default.**
    - **Recent ID issuance & address mismatches should trigger additional scrutiny.**
4. **Geographical & Employment Mismatch:**
    - **Clients living far from their work location are riskier.**
    - **Consider verifying job stability before approval.**
5. **Loan & Annuity Optimization:**
    - **Higher loan annuity generally aligns with higher income → Lowers default risk.**

This report provides valuable insights into loan application trends and default risks, aiding in better financial decision-making.

Link to clean excel file

Link to Google Colab