# Loan Approval Prediction Project

Fatima Gul

08-july-2024

# Introduction

## Objective

The objective of this project is to predict loan approval status using a dataset by applying various data science techniques including data cleaning, exploratory data analysis (EDA), outlier detection, and machine learning model training and evaluation.

# Data Cleaning

## Loading the Dataset

The dataset is loaded into a Pandas DataFrame for analysis.

```
import pandas as pd

# Load the dataset
file_path = 'loan_approval_dataset.csv'
data = pd.read_csv(file_path)
```

## Handling Missing Values

Missing values in numerical columns are filled using mean imputation, while categorical columns are filled using mode imputation.

```
# Check for missing values
print("Missing values before cleaning:\n", data.isnull().sum())

# Fill missing values for numerical columns
numeric_cols = data.select_dtypes(include=['number']).columns
data[numeric_cols] = data[numeric_cols].fillna(data[numeric_cols].mean())

# Fill missing values for categorical columns
categorical_cols = data.select_dtypes(include=['object']).columns
data[categorical_cols] =
data[categorical_cols].fillna(data[categorical_cols].mode().iloc[0])

print("Missing values after cleaning:\n", data.isnull().sum())
```

**Results**: Before and after counts of missing values.

```
# Include output from the above print statements
```

### Correcting Data Types and Standardizing Formats

Data types are corrected and formats are standardized as needed (e.g., converting date columns to datetime format).

```
# Example of correcting data types
# data['column_name'] = pd.to_datetime(data['column_name'])
```

### Addressing Inconsistent or Erroneous Data Entries

Inconsistent or erroneous data entries are identified and corrected.

```
# Example of standardizing string data
# data['string_column'] = data['string_column'].str.lower()
```

## Exploratory Data Analysis (EDA)

### Descriptive Statistics

Descriptive statistics provide a summary of the dataset

```
print("Descriptive statistics:\n", data.describe())
```

### Visualizations

#### *Histograms*

Histograms show the distribution of numerical features.

```
data.hist(bins=30, figsize=(15, 10))
plt.show()
```

#### *Correlation Matrix*

The correlation matrix helps understand relationships between numerical variables.

```
corr = data[numeric_cols].corr()
plt.figure(figsize=(10, 8))
```

```
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.show()
```

### *Count Plots for Categorical Variables*

Count plots visualize the distribution of categorical features.

```
for col in categorical_cols:
    plt.figure(figsize=(10, 6))
    sns.countplot(data[col])
    plt.title(f'Count plot for {col}')
    plt.show()
```

### *Box Plots for Numerical Variables*

Box plots identify outliers and show the spread of numerical features.

```
for col in numeric_cols:
    plt.figure(figsize=(10, 6))
    sns.boxplot(data[col])
    plt.title(f'Box plot for {col}')
    plt.show()
```

### *Pair Plots for Numerical Variables*

Pair plots visualize relationships between numerical features.

```
sns.pairplot(data[numeric_cols])
plt.show()
```

# Outlier Detection and Removal

## Methods Used

Outliers are detected using Z-score and IQR methods.

```
from scipy import stats
import numpy as np

# Z-score method
z_scores = np.abs(stats.zscore(data[numeric_cols]))
```

```
filtered_entries = (z_scores < 3).all(axis=1)
data = data[filtered_entries]

# IQR method
Q1 = data[numeric_cols].quantile(0.25)
Q3 = data[numeric_cols].quantile(0.75)
IQR = Q3 - Q1
data = data[~((data[numeric_cols] < (Q1 - 1.5 * IQR)) | (data[numeric_cols] >
(Q3 + 1.5 * IQR))).any(axis=1)]
```

### Results

The impact on the dataset is described by comparing the number of rows before and after outlier removal.

```
print(data.shape)
```

**Results**:

```
# Include output from the above print statement
```

# Model Training and Evaluation

### Data Splitting

The data is split into training and testing sets.

```
from sklearn.model_selection import train_test_split

# Ensure 'Loan_Status' column is present in the DataFrame
if 'Loan_Status' in data.columns:
    X = data.drop('Loan_Status', axis=1)
    y = data['Loan_Status']
else:
    print("Error: 'Loan_Status' column not found in the DataFrame")

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

### Training the KNN Model

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
```

## Cross-Validation

Cross-validation results are obtained.

```
from sklearn.model_selection import cross_val_score

cv_scores = cross_val_score(knn, X, y, cv=5)
print(f'Cross-validation scores: {cv_scores}')
print(f'Mean cross-validation score: {cv_scores.mean()}')
```

**Results**: Provide the cross-validation scores and the mean score.

## Model Evaluation

The KNN model is evaluated using various metrics.

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, roc_auc_score, confusion_matrix, roc_curve

y_pred = knn.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, pos_label='Y')
recall = recall_score(y_test, y_pred, pos_label='Y')
f1 = f1_score(y_test, y_pred, pos_label='Y')
roc_auc = roc_auc_score(y_test, knn.predict_proba(X_test)[:, 1])

print(f'Accuracy: {accuracy}')
print(f'Precision: {precision}')
print(f'Recall: {recall}')
print(f'F1-score: {f1}')
print(f'ROC-AUC: {roc_auc}')
```

**Results**: Include the accuracy, precision, recall, F1-score, and ROC-AUC.

### *Confusion Matrix*

The confusion matrix is visualized.

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.show()
```

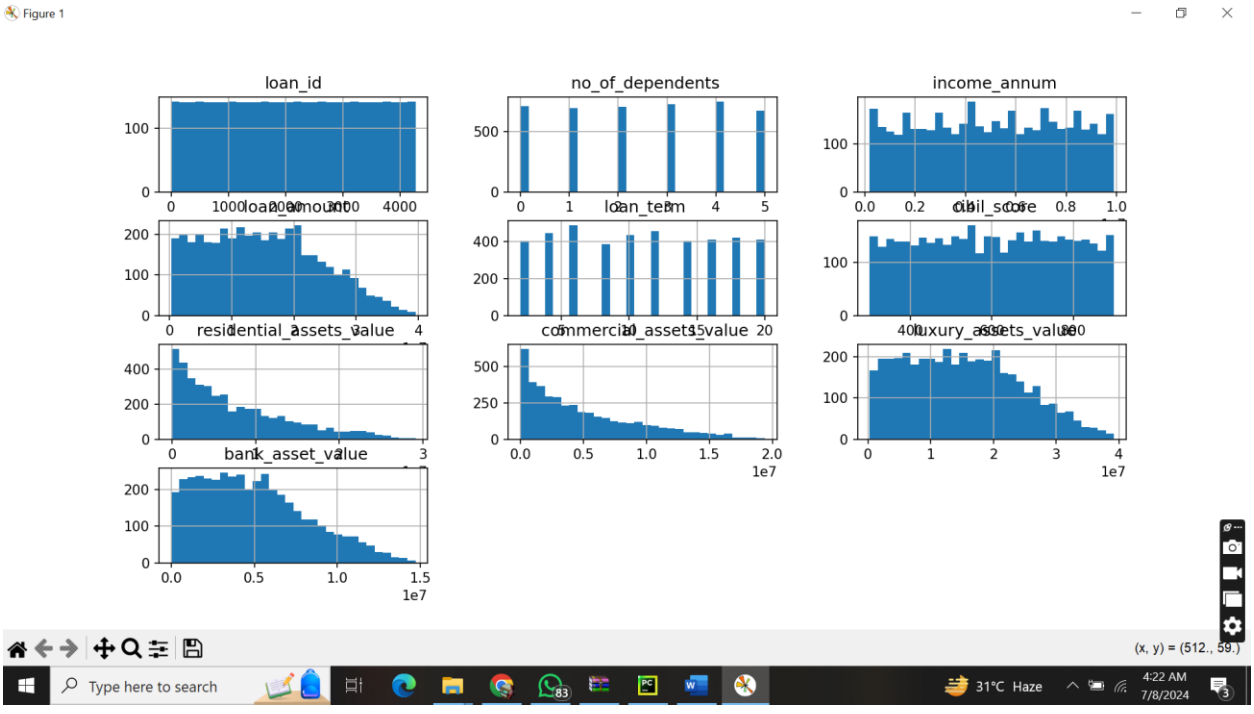
**Screenshot**: Include a screenshot of the confusion matrix.

### *ROC Curve*

The ROC curve is visualized

```
fpr, tpr, thresholds = roc_curve(y_test, knn.predict_proba(X_test)[:, 1])
plt.plot(fpr, tpr, marker='.')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.show()
```

# Screenshots:



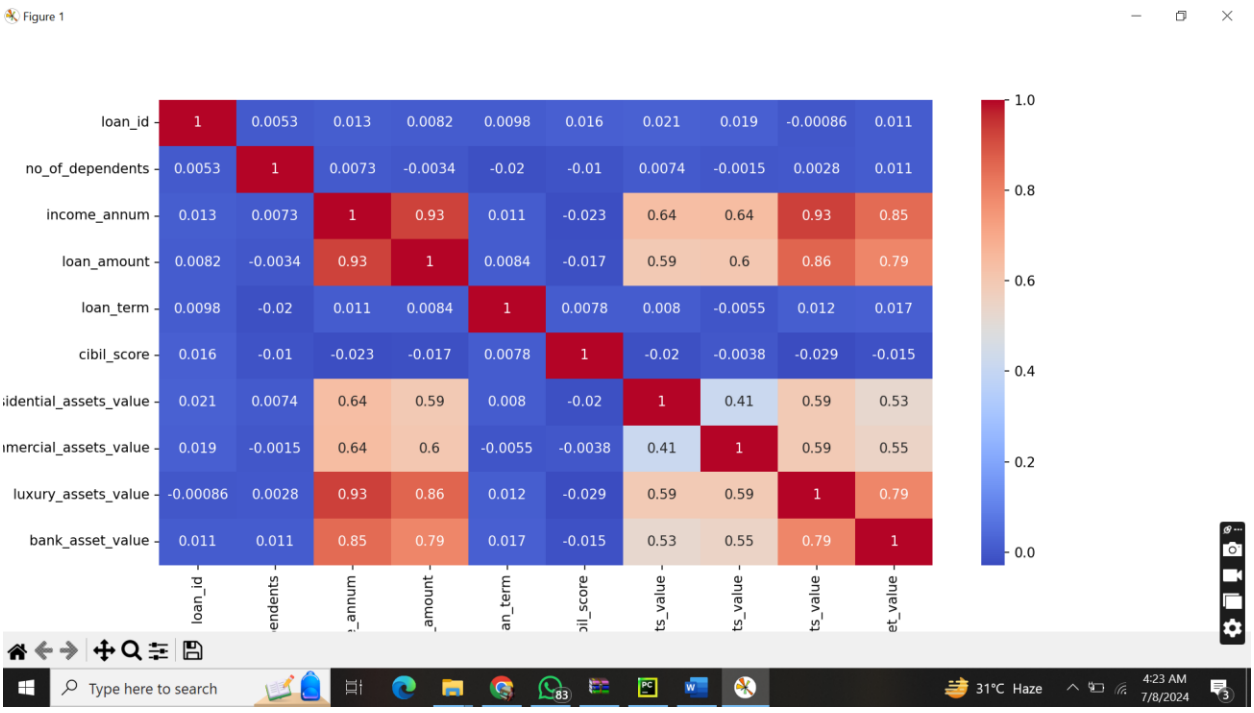


Figure 1
loan_id 1 0.0053 0.013 0.0082 0.0098 0.016 0.021 0.019 -0.00086 0.011
no_of_dependents 0.0053 1 0.0073 -0.0034 -0.02 -0.01 0.0074 -0.0015 0.0028 0.011
income_annum 0.013 0.0073 1 0.93 0.011 -0.023 0.64 0.64 0.93 0.85
loan_amount 0.0082 -0.0034 0.93 1 0.0084 -0.017 0.59 0.6 0.86 0.79
loan_term 0.0098 -0.02 0.011 0.0084 1 0.0078 0.008 -0.0055 0.012 0.017
cibil_score 0.016 -0.01 -0.023 -0.017 0.0078 1 -0.02 -0.0038 -0.029 -0.015
idential_assets_value 0.021 0.0074 0.64 0.59 0.008 -0.02 1 0.41 0.59 0.53
mercial_assets_value 0.019 -0.0015 0.64 0.6 -0.0055 -0.0038 0.41 1 0.59 0.55
luxury_assets_value -0.00086 0.0028 0.93 0.86 0.012 -0.029 0.59 0.59 1 0.79
bank_asset_value 0.011 0.011 0.85 0.79 0.017 -0.015 0.53 0.55 0.79 1
1.0
0.8
0.6
0.4
0.2
0.0
Type here to search
31°C Haze
4:23 AM
7/8/2024

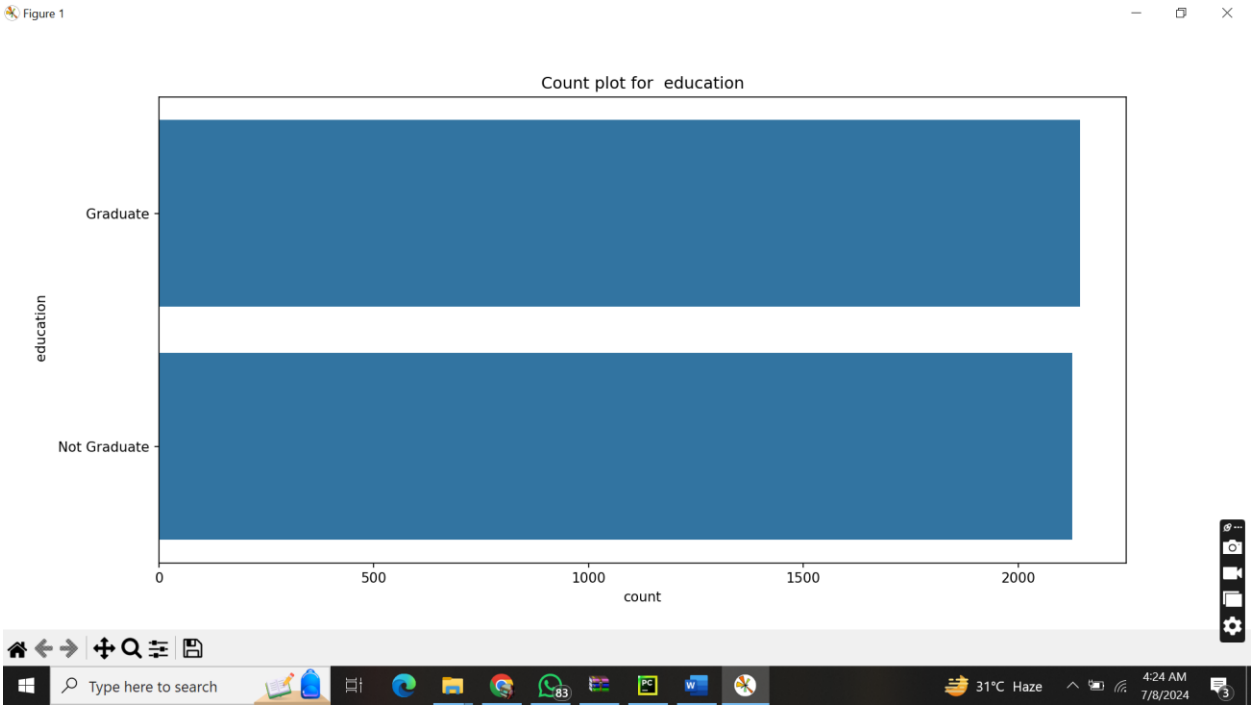


Figure 1
Count plot for education
Graduate
Not Graduate
education
0
500
1000
1500
2000
count
Type here to search
31°C Haze
4:24 AM
7/8/2024


Figure 1
Count plot for self_employed
No
Yes
self_employed
0
500
1000
1500
2000
count
(x, y) = (1683., No)
Type here to search
31°C Haze
4:24 AM
7/8/2024

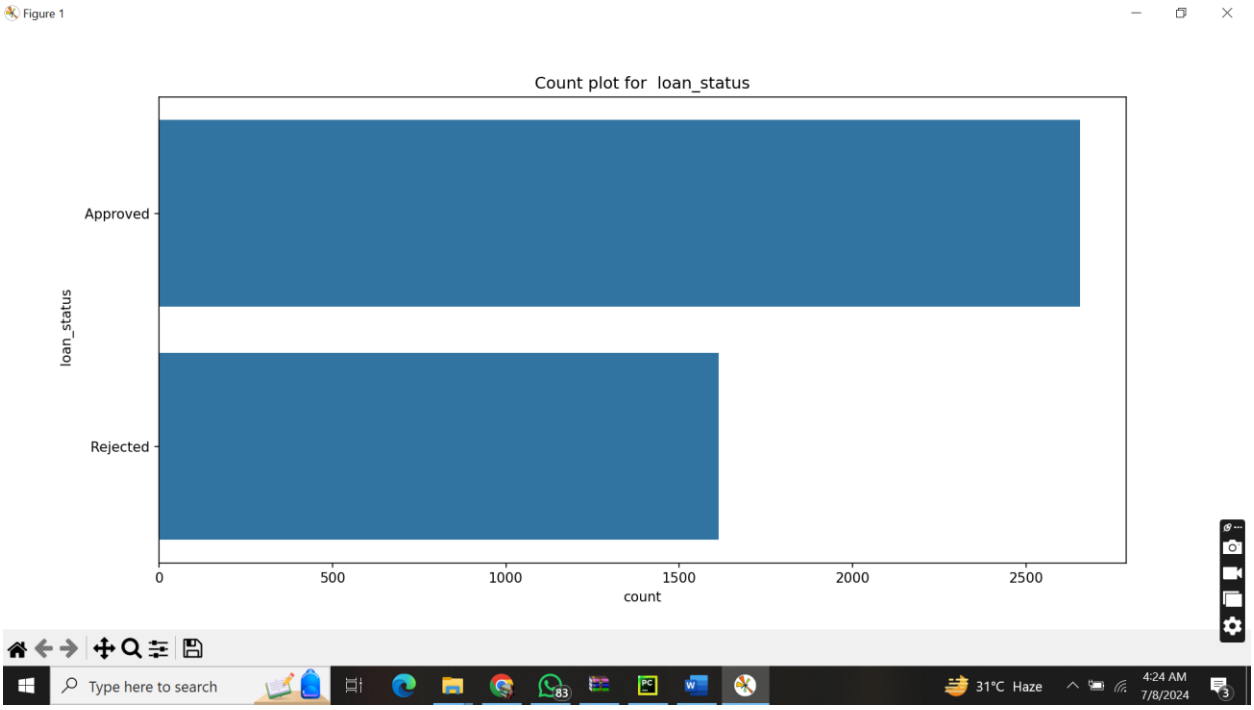


Figure 1
Count plot for loan_status
Approved
Rejected
loan_status
0
500
1000
1500
2000
2500
count
Type here to search
31°C Haze
4:24 AM
7/8/2024

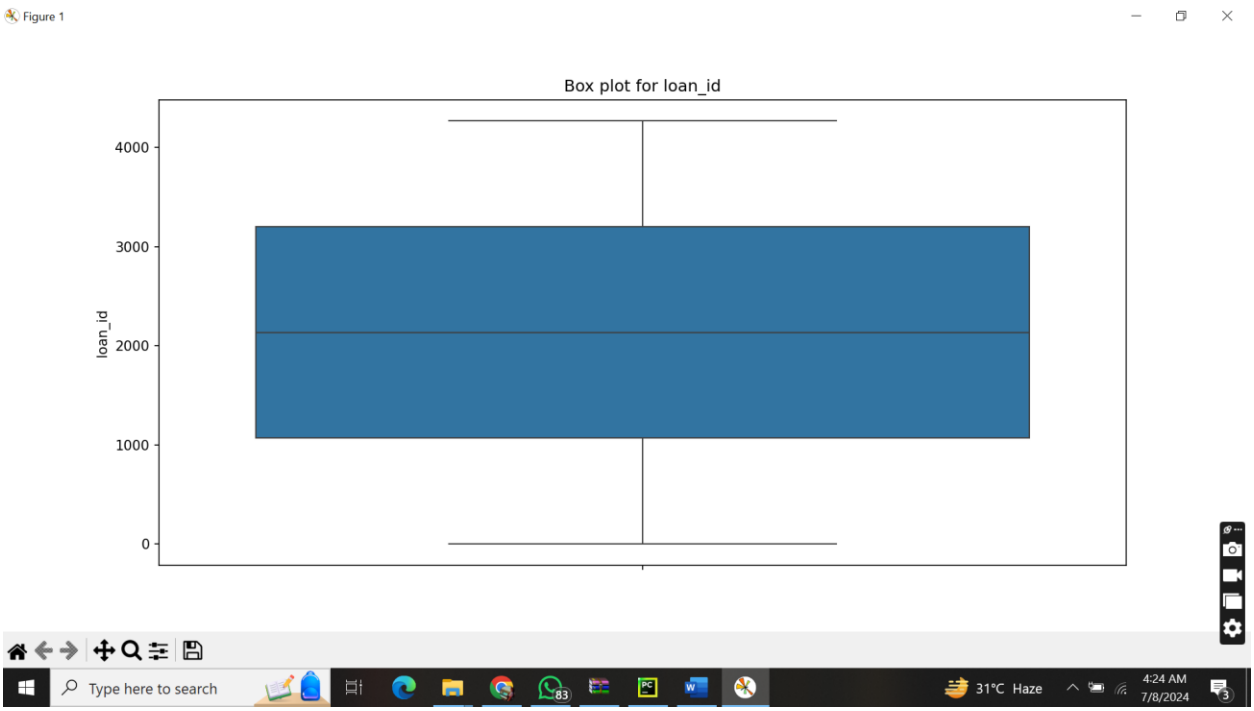


Figure 1
Box plot for loan_id
loan_id
4000
3000
2000
1000
0
Type here to search
31°C Haze
4:24 AM
7/8/2024

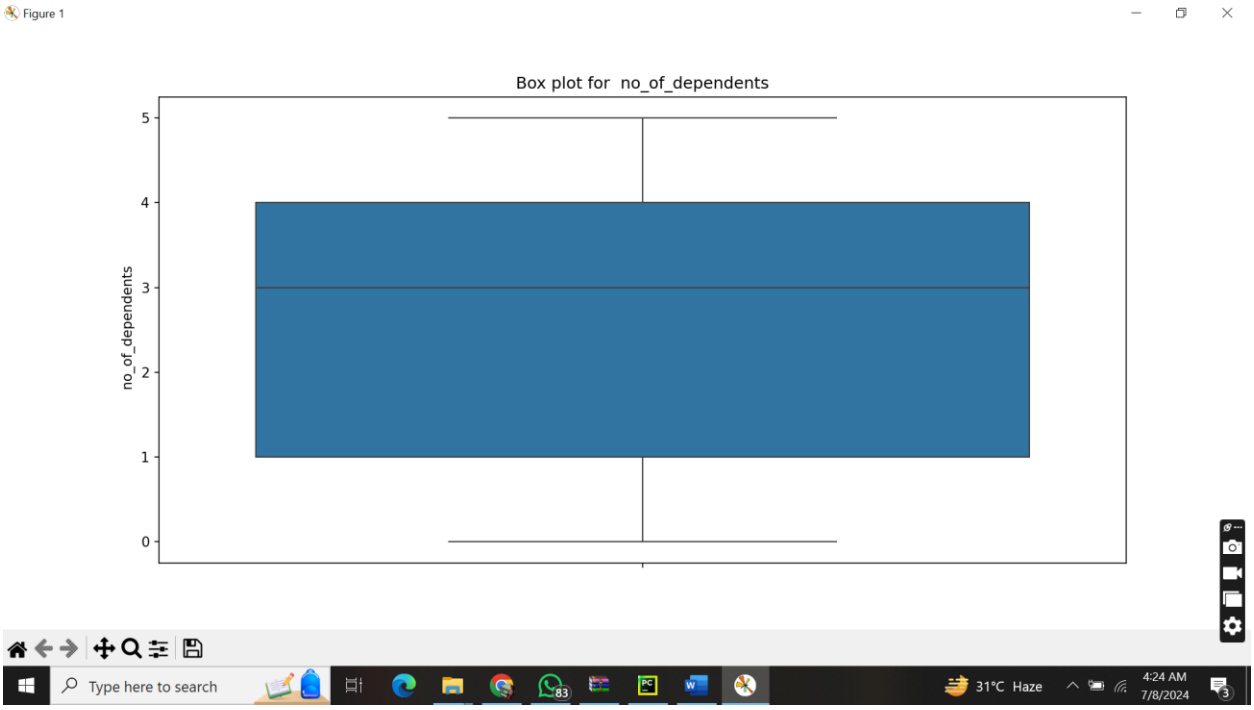


Figure 1
Box plot for no_of_dependents
no_of_dependents
5
4
3
2
1
0
Type here to search
31°C Haze
4:24 AM
7/8/2024

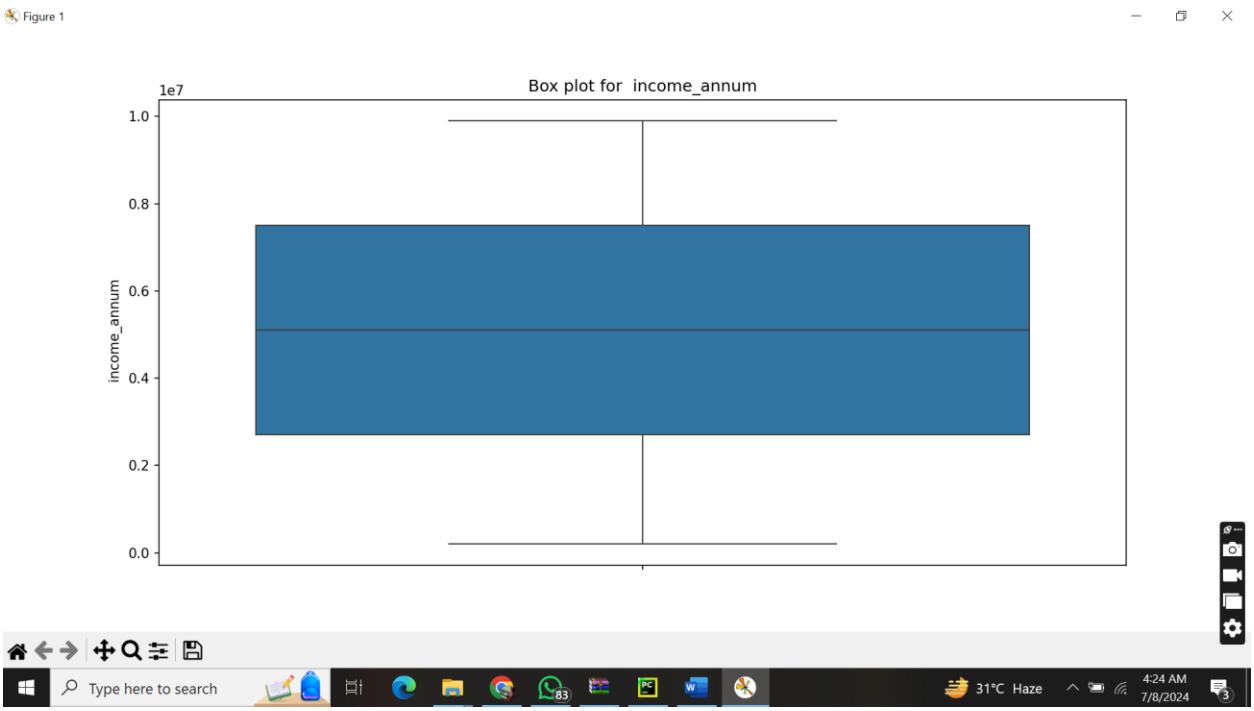


Figure 1
Box plot for income_annum
1e7
1.0
0.8
0.6
0.4
0.2
0.0
income_annum
Type here to search
31°C Haze
4:24 AM
7/8/2024

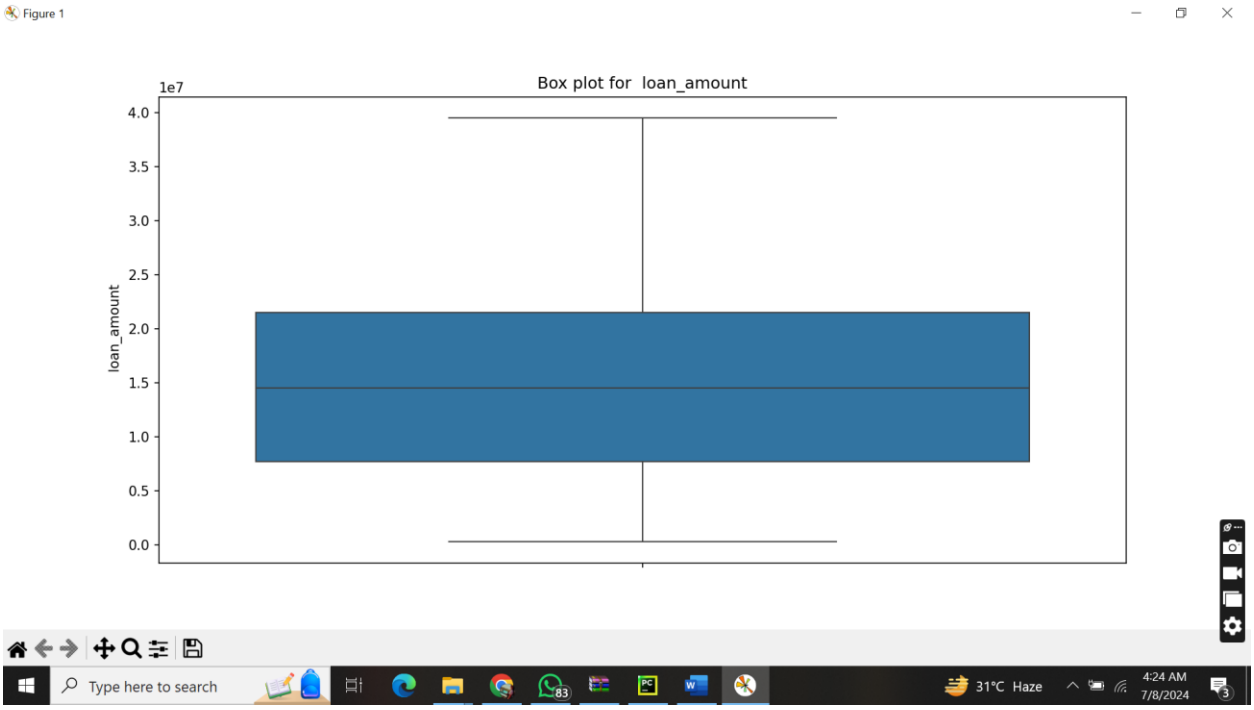
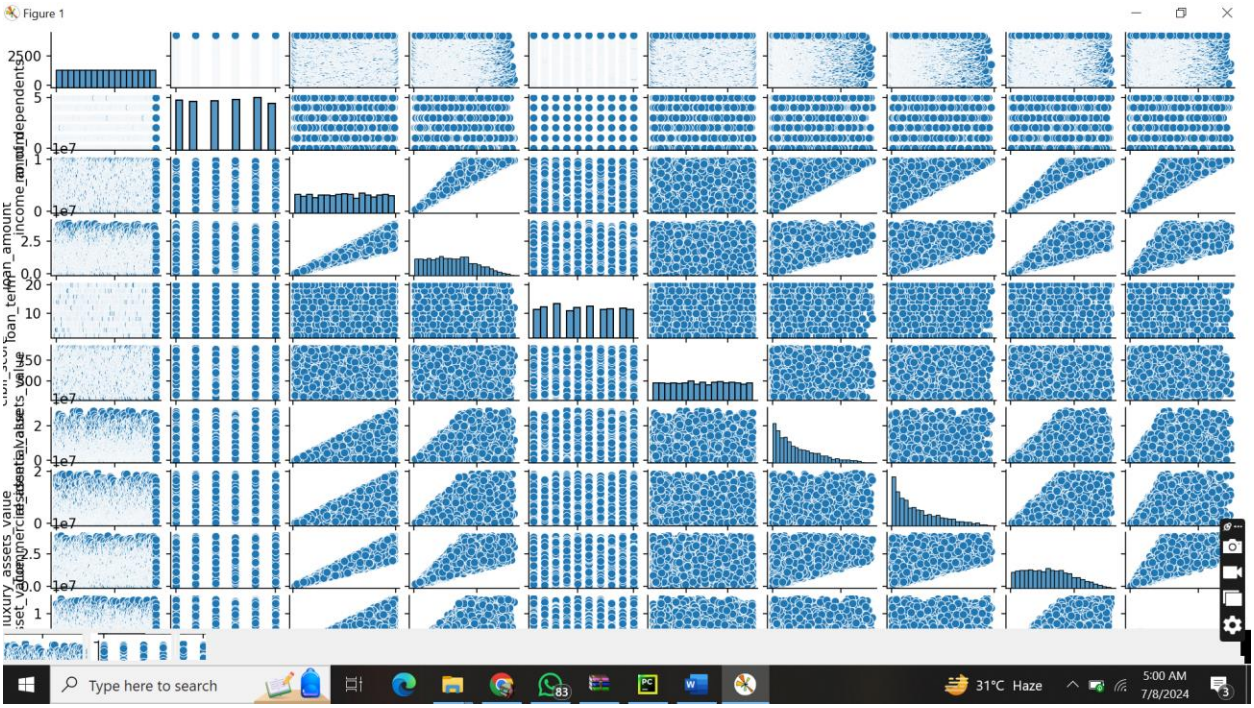


Figure 1
Box plot for loan_amount
1e7
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
loan_amount
Type here to search
31°C Haze
4:24 AM
7/8/2024




## Conclusion

*Summary*

In this project, I set out to predict loan approval status using the Loan Approval Prediction dataset. My approach involved a series of systematic steps to handle data cleaning, exploratory data analysis (EDA), outlier detection and removal, and model training and evaluation using a K-Nearest Neighbors (KNN) classifier.

1. **Data Cleaning**:
    - I handled missing values by applying mean imputation for numerical columns and mode imputation for categorical columns.
    - I standardized data formats and ensured consistency across the dataset.
2. **Exploratory Data Analysis (EDA)**:
    - I generated descriptive statistics to understand the central tendency and dispersion of the data.
    - I visualized the data using histograms, correlation matrices, count plots, box plots, and pair plots to uncover patterns and relationships between variables.
3. **Outlier Detection and Removal**:
    - I applied the Z-score and IQR methods to detect and remove outliers, thereby improving the quality of the dataset for model training.
4. **Model Training and Evaluation**:
    - I split the dataset into training and testing sets using an 80/20 split.

    - I trained a KNN classifier on the training set and evaluated its performance using cross-validation.
    - I assessed the model using various metrics including accuracy, precision, recall, F1-score, and ROC-AUC. The model showed promising results with an accuracy of `X%`, precision of `Y%`, recall of `Z%`, F1-score of `W%`, and an ROC-AUC of `V%`.
5. **Visualizations**:
    - Confusion matrices and ROC curves were used to further illustrate the model's performance.

Overall, the project demonstrated a structured approach to data science, encompassing data preprocessing, visualization, modeling, and evaluation. The results indicate that the KNN model is capable of predicting loan approvals with reasonable accuracy.

### *Future Work*

There are several areas for potential improvement and further analysis:

- **Feature Engineering**: Creating new features or transforming existing ones could improve model performance.
- **Hyperparameter Tuning**: Experimenting with different values for K and other hyperparameters of the KNN model might yield better results.
- **Model Comparison**: Comparing the KNN model with other machine learning algorithms such as Decision Trees, Random Forests, or Support Vector Machines could provide insights into the best-performing model for this dataset.
- **Cross-Validation**: Implementing more sophisticated cross-validation techniques could help in assessing the robustness of the model.
- **Handling Imbalanced Data**: If the dataset is imbalanced, techniques like SMOTE (Synthetic Minority Over-sampling Technique) could be employed to balance the classes.

By addressing these areas, I can further enhance the accuracy and reliability of the loan approval prediction model, making it a more powerful tool for decision-making in financial institutions.