Implementation of Logistic Regression for Loan Approval Classification

Credit Risk Analysis and Modeling Using Categorical and Continuous Variables in a Synthetic Dataset.

Hernández Pérez Erick Fernando

Figure 1

In the financial field, credit risk assessment is a fundamental process for banking and lending institutions. Loan approval classification allows determining the probability that an applicant will meet his or her payment obligations, thus minimizing the risks associated with default and optimizing the allocation of financial resources. To address this challenge, predictive modeling techniques are used that allow informed decisions to be made based on historical data and relevant characteristics of the applicants.

Among the various classification techniques, logistic regression has established itself as one of the most widely used methodologies due to its ability to model relationships between explanatory variables and a binary target variable, in this case, the approval or rejection of a loan. Its intuitive interpretation and computational efficiency make it a valuable tool in financial analysis.

In this study, a logistic regression model will be implemented using the Loan Approval Classification Data dataset by Lo (2024)[2], which serves as the foundation for the analysis. This dataset incorporates both categorical and continuous variables that influence the loan approval decision and has been enriched using the SMOTENC technique to balance the representation of the classes and enhance the model's predictive capacity. Additionally, the dataset was further enriched with the Credit Risk Dataset by Lao Tse (2024)[1], which provides original data on credit risk, as well as with variables related to financial risk for loan approval from the Financial Risk for Loan Approval dataset by Zoppelletto (2024)[3]. The combination of these datasets allowed the creation of a more robust and comprehensive dataset, improving the model's ability to classify loan applications. Through this analysis, we aim not only to develop a predictive model but also to understand the key factors influencing the classification and evaluate the model's performance in terms of accuracy and generalization capacity.

Our dataset

The dataset used in this study consists of 45,000 records and 14 variables, each providing valuable information about loan applicants and their loan applications. The columns and their descriptions are as follows:

Column	Description	Type
person_age	Age of the person	Float
person_gender	Gender of the person	Categorical
person_education	Highest level of education achieved by the person	Categorical
person_income	Annual income of the person	Float
person_emp_exp	Years of employment experience	Integer
person_home_ownership	Home ownership status, such as rent, own, or mortgage	Categorical
loan_amnt	The loan amount requested by the person	Float
loan_intent	The purpose of the loan	Categorical
loan_int_rate	The interest rate of the loan	Float
loan_percent_income	The loan amount as a percentage of the applicant's annual income	Float
cb_person_cred_hist_length	Length of the applicant's credit history in years	Float
credit_score	Credit score of the person	Integer
previous_loan_defaults_on_file	Indicator of any previous loan defaults on file	Categorical
loan_status	Target variable indicating the loan approval status; 1 = approved, 0 = rejected	Integer

This dataset provides a comprehensive set of features related to both the applicant and their loan, allowing for detailed analysis and modeling of the loan approval decision-making process.

Let's get started

Loading the dataset

```
import kagglehub
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve, auc

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Download the dataset using kagglehub
path = kagglehub.dataset_download("taweilo/loan-approval-classification-data")

# Verify the downloaded path
print("Path to dataset files:", path)

# List the files inside the downloaded directory
files = os.listdir(path)
print("Downloaded files:", files)
```

```python
# Search for the CSV file in the folder
csv_files = [f for f in files if f.endswith('.csv')]
if not csv_files:
    raise FileNotFoundError("No CSV file was found in the downloaded folder.")

# Load the CSV file into a DataFrame
csv_path = os.path.join(path, csv_files[0])  # Take the first CSV file found
df = pd.read_csv(csv_path)

# Display the first rows
print(df.head())
```

```
Path to dataset files: /root/.cache/kagglehub/datasets/taweilo/loan-approval-classification-data/versions/1
Downloaded files: ['loan_data.csv']
   person_age person_gender person_education  person_income  person_emp_exp  \
0        22.0        female           Master        71948.0               0
1        21.0        female      High School        12282.0               0
2        25.0        female      High School        12438.0               3
3        23.0        female         Bachelor        79753.0               0
4        24.0          male           Master        66135.0               1

  person_home_ownership  loan_amnt loan_intent  loan_int_rate  \
0                  RENT    35000.0    PERSONAL          16.02
1                   OWN     1000.0   EDUCATION          11.14
2              MORTGAGE     5500.0     MEDICAL          12.87
3                  RENT    35000.0     MEDICAL          15.23
4                  RENT    35000.0     MEDICAL          14.27

   loan_percent_income  cb_person_cred_hist_length  credit_score  \
0                 0.49                         3.0           561
1                 0.08                         2.0           504
2                 0.44                         3.0           635
3                 0.44                         2.0           675
4                 0.53                         4.0           586

  previous_loan_defaults_on_file  loan_status
0                             No            1
1                            Yes            0
2                             No            1
3                             No            1
4                             No            1
```

Figure 2

Loading the dataset

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45000 entries, 0 to 44999
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   person_age                      45000 non-null  float64
 1   person_gender                   45000 non-null  object
 2   person_education                45000 non-null  object
 3   person_income                   45000 non-null  float64
 4   person_emp_exp                  45000 non-null  int64
 5   person_home_ownership           45000 non-null  object
 6   loan_amnt                       45000 non-null  float64
 7   loan_intent                     45000 non-null  object
 8   loan_int_rate                   45000 non-null  float64
 9   loan_percent_income             45000 non-null  float64
 10  cb_person_cred_hist_length      45000 non-null  float64
 11  credit_score                    45000 non-null  int64
 12  previous_loan_defaults_on_file  45000 non-null  object
 13  loan_status                     45000 non-null  int64
dtypes: float64(6), int64(3), object(5)
memory usage: 4.8+ MB
```

Figure 3

```
df.describe(include='all')
```

	person_age	person_gender	person_education	person_income	person_emp_exp	person_home_ownership	loan_amnt
count	45000.0	45000	45000	45000.0	45000.0	45000	45000.0
unique	nan	2	5	nan	nan	4	nan
top	nan	male	Bachelor	nan	nan	RENT	nan
freq	nan	24841	13399	nan	nan	23443	nan
mean	27.764177777777778	nan	nan	80319.05322222222	5.410333333333333	nan	9583.157555555556
std	6.045108211348622	nan	nan	80422.49863189556	6.063532086575209	nan	6314.886690541405
min	20.0	nan	nan	8000.0	0.0	nan	500.0
25%	24.0	nan	nan	47204.0	1.0	nan	5000.0
50%	26.0	nan	nan	67048.0	4.0	nan	8000.0
75%	30.0	nan	nan	95789.25	8.0	nan	12237.25
max	144.0	nan	nan	7200766.0	125.0	nan	35000.0

Figure 4

loan_intent	loan_int_rate	loan_percent_income	cb_person_cred_hist_length	credit_score	previous_loan_defaults_on_file	loan_status
45000	45000.0	45000.0	45000.0	45000.0	45000	45000.0
6	nan	nan	nan	nan	2	nan
EDUCATION	nan	nan	nan	nan	Yes	nan
9153	nan	nan	nan	nan	22858	nan
nan	11.006605777777779	0.1397248888888889	5.867488888888885	632.6087555555556	nan	0.2222222222222222
nan	2.9788082802254734	0.08721230801403355	3.8797018451620433	50.435865000741984	nan	0.41574432904844355
nan	5.42	0.0	2.0	390.0	nan	0.0
nan	8.59	0.07	3.0	601.0	nan	0.0
nan	11.01	0.12	4.0	640.0	nan	0.0
nan	12.99	0.19	8.0	670.0	nan	0.0
nan	20.0	0.66	30.0	850.0	nan	1.0

Figure 5

```python
# Clean data
df = df.loc[df['person_age'] <= 90]

# Identify columns by type
num_cols = df.select_dtypes(include=['float64', 'int64']).columns
cat_cols = df.select_dtypes(include=['object']).columns

# Set figure size
plt.figure(figsize=(15, 8))

# Plot numerical variables using histograms
for i, col in enumerate(num_cols, 1):
    plt.subplot(3, 3, i)  # Adjust based on the number of numerical columns
    sns.histplot(df[col], kde=True, bins=30)
    plt.title(f"Distribution of {col}")

plt.tight_layout()
plt.show()

# Plot categorical variables using bar charts
plt.figure(figsize=(15, 8))

for i, col in enumerate(cat_cols, 1):
    plt.subplot(2, 3, i)  # Adjust based on the number of categorical columns
    sns.countplot(y=df[col], order=df[col].value_counts().index)
    plt.title(f"Count of {col}")

plt.tight_layout()
plt.show()
```

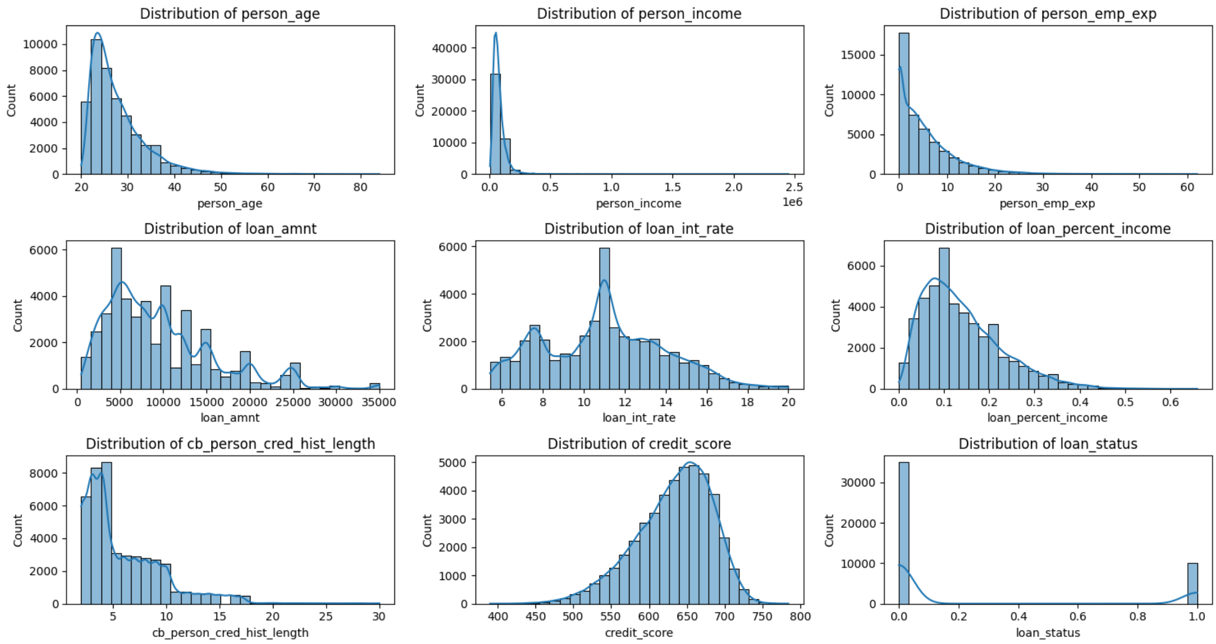

Figure 6

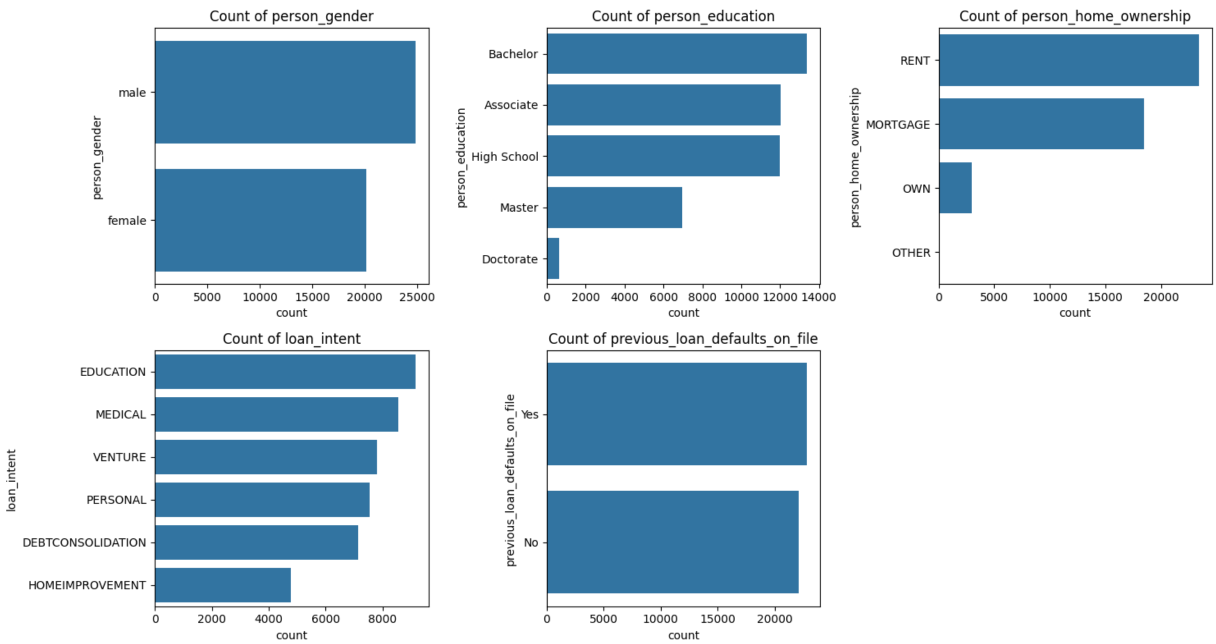

Figure 7

```
df['person_gender'] = df['person_gender'].map({'male': 0, 'female': 1})
df['person_education'] = df['person_education'].map({'Bachelor': 0, 'Associate': 1, 'High
↪  School': 2, 'Master': 3, 'Doctorate': 4})
df['person_home_ownership'] = df['person_home_ownership'].map({'RENT': 0, 'MORTGAGE': 1,
↪  'OWN': 2, 'OTHER': 3})
df['loan_intent'] = df['loan_intent'].map({'EDUCATION': 0, 'MEDICAL': 1, 'VENTURE': 2,
↪  'PERSONAL': 3, 'DEBTCONSOLIDATION': 4, 'HOMEIMPROVEMENT': 5})
```

```
df['previous_loan_defaults_on_file'] = df['previous_loan_defaults_on_file'].map({'No': 0,
↪   'Yes': 1})
```

The correlation matrix

```
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(),annot=True,fmt=".3f", linewidth=.5)
```

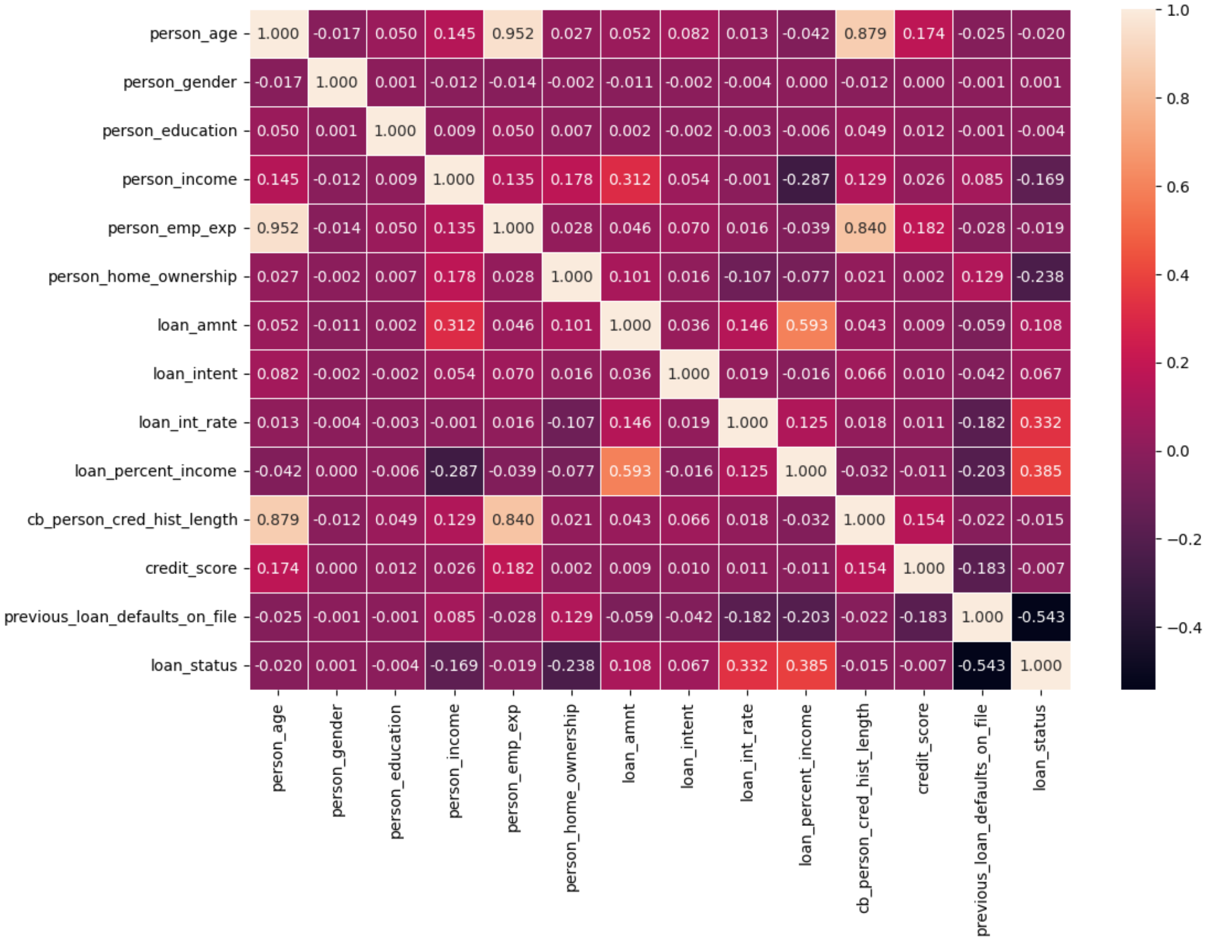

Figure 8

The model

```
features = [col for col in df.corr().index[abs(df.corr()['loan_status']) >= 0.01] if col
↪   != 'loan_status']

X = df[features]
y = df['loan_status']

scaler = StandardScaler()
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns, index=X.index)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.15,
↪   random_state=42)
```

```
model = keras.Sequential([
    layers.Dense(1, activation='sigmoid', input_shape=(X_train.shape[1],))
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.fit(X_train, y_train, epochs=30, batch_size=32, validation_data=(X_test, y_test))
```

The confusion matrix

```
# Get probability predictions
y_probs = model.predict(X_test)

# Convert probabilities to binary predictions (0 or 1)
y_pred = (y_probs > 0.4).astype(int)

plt.figure(figsize=(6,5))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", linewidths=0.5)

# Etiquetas de los ejes
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.xticks(ticks=[0.5, 1.5], labels=["Class 0", "Class 1"])
plt.yticks(ticks=[0.5, 1.5], labels=["Class 0", "Class 1"], rotation=0)

plt.show()
```

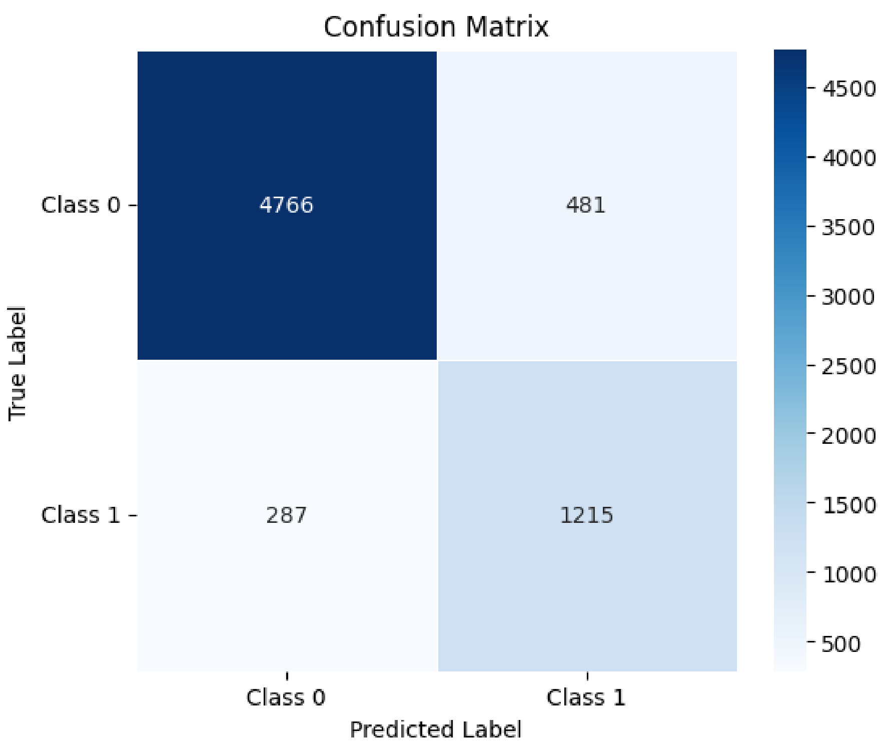

Figure 9

```python
TN, FP, FN, TP = cm.ravel()

Accuracy = (TP+TN) / (TP+TN+FP+FN)
TPR = TP / (TP + FN)  # True Positive Rate
FPR = FP / (FP + TN)  # False Positive Rate
Precision = TP / (TP + FP)

print(f"Accuracy: {Accuracy:.4f}")
print(f"True Positive Rate (FPR): {TPR:.4f}")
print(f"False Positive Rate (FPR): {FPR:.4f}")
print(f"Precision: {Precision:.4f}")
```

```
Accuracy: 0.8918
True Positive Rate (FPR): 0.7304
False Positive Rate (FPR): 0.0619
Precision: 0.7714
```

Figure 10

The ROC Curve

```python
fpr, tpr, _ = roc_curve(y_test, y_probs)

# Compute AUC (Area Under the Curve)
roc_auc = auc(fpr, tpr)

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label=f'ROC Curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')  # Random classifier line
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```

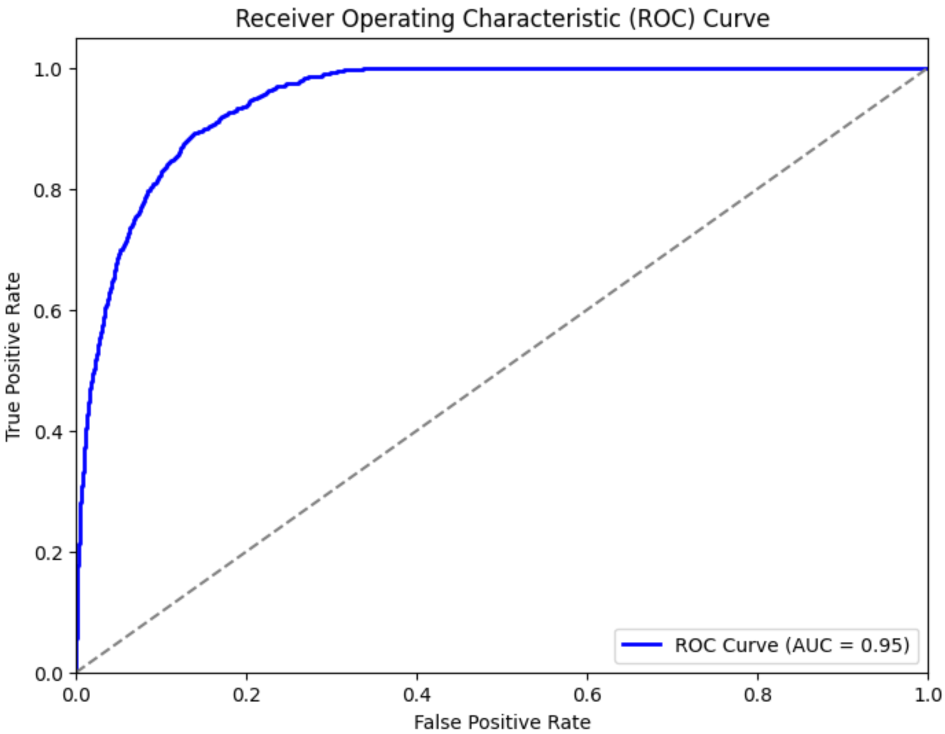

Figure 11

Conclusions

In this study, a logistic regression model was developed using a dataset that includes both categorical and continuous variables. The model was built based on features selected through correlation analysis, which included the following attributes: `person_age`, `person_income`, `person_emp_exp`, `person_home_ownership`, `loan_amnt`, `loan_intent`, `loan_int_rate`, `loan_percent_income`, `cb_person_cred_hist_length`, and `previous_loan_defaults_on_file`. These features were found to be highly relevant for predicting loan approval status, and their inclusion helped create a robust model.

With a threshold of 0.4, the model achieved the following performance metrics:

- **Accuracy**: 0.9818, which indicates that the model correctly classified 89.18% of loan applications.
- **True Positive Rate (TPR)**: 0.7304, demonstrating that the model successfully identified 73.04% of the actual loan approvals.
- **False Positive Rate (FPR)**: 0.0619, meaning that the model had a low rate of incorrectly classifying rejected loans as approved.
- **Precision**: 0.7714, showing that 77.14% of the loans classified as approved by the model were actually approved.

Additionally, the model's **ROC curve** showed an **AUC of 0.95**, which is indicative of excellent model performance. An AUC score closer to 1.0 suggests that the model is highly effective at distinguishing between loan approvals and rejections.

These results suggest that the logistic regression model is effective at predicting loan approval decisions. The model demonstrates high accuracy, precision, and a strong ability to correctly identify both true positives and true negatives, making it a reliable tool for loan approval classification. However, further optimization and testing with different thresholds and techniques could be explored to improve the model's generalization capacity.

References

[1] Lao Tse. (2024). Credit risk dataset [Dataset]. Kaggle. https://www.kaggle.com/datasets/laotse/credit-risk-dataset

[2] Lo, T. (2024). Loan approval classification data [Dataset]. Kaggle. https://www.kaggle.com/datasets/taweilo/loan-approval-classification-data

[3] Zoppelletto, L. (2024). Financial risk for loan approval [Dataset]. Kaggle. https://www.kaggle.com/datasets/lorenzozoppelletto/financial-risk-for-loan-approval