Credit Worthiness

Optimization of Machine Learning Models

Leor Seal, Elena Tarassenko, Ziye Jin, David Gamarra

Group Delegation

- David - Data Exploration & Database Access
- Ziye - Data Cleaning
- Leor - Model Development
- Elena - Model Optimization

CONTENTS

01 | CASE

Credit Worthiness

Case

As data scientists employed by a global finance corporation, our responsibilities involve handling a vast repository of essential banking data, particularly a substantial collection of credit-related information. Leadership has expressed an interest in developing an intelligent system that can categorize individuals into specific credit score categories, thereby streamlining and minimizing manual workload. The modeling techniques are significant in real world scenarios, as credit scores help determine mortgage rates offered by banks, among other financial rates.

02

Backend

Sourcing, Data Wrangling

Database Access

- Database was created from dataset to simulated real-world scenario.
- Once access to database is obtained via **sqlalchemy**, we can then create dataframe using **pandas**
- Data is then onboarded to the cleaning phase

```
import pandas as pd
from sqlalchemy import create_engine

# Replace 'sqlite:///credit_worthiness.db' with the correct database URL for SQLite
engine = create_engine('sqlite:///C:\\Users\\KING\\Documents\\bootcamp_challenges\\Project-4-Credit-Worthiness\\Reso

# Specify the table names
train_table_name = 'TrainTable'
test_table_name = 'TestTable'

# Query data and create DataFrames
train_query = f'SELECT * FROM {train_table_name};'
train_df = pd.read_sql_query(train_query, engine)

test_query = f'SELECT * FROM {test_table_name};'
test_df = pd.read_sql_query(test_query, engine)
```

Python

```
# Display the DataFrames
print("Train DataFrame:")
train_df
```

Python

ain DataFrame:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Salary	Num_Bank_A
0	0x1602	CUS_0xd40	January	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	1824.843333	
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	NaN	
2	0x1604	CUS_0xd40	March	Aaron Maashoh	-500	821-00-0265	Scientist	19114.12	NaN	

Credit Score Classification Dataset Information

Given a person's credit-related information, build a machine learning model that can classify the credit score.

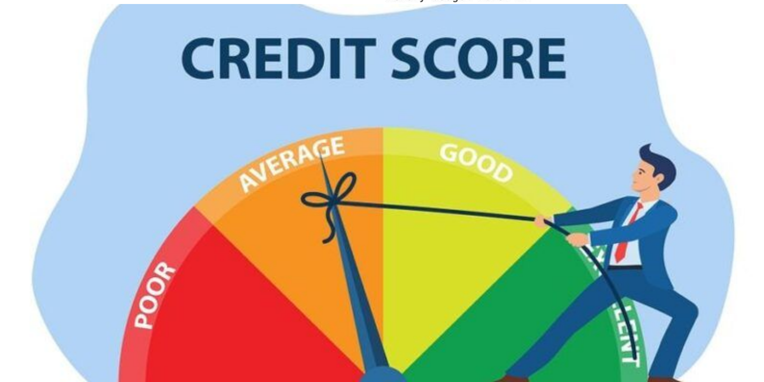
Credit-related information:

Train.csv: 27 columns and 100,000 Unique Values

Test.csv: 27 columns and 50,000 Unique Values

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 28 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   ID                        100000 non-null  object 
 1   Customer_ID               100000 non-null  object 
 2   Month                     100000 non-null  object 
 3   Name                      90015 non-null   object 
 4   Age                       100000 non-null  object 
 5   SSN                       100000 non-null  object 
 6   Occupation                100000 non-null  object 
 7   Annual_Income             100000 non-null  object 
 8   Monthly_Inhand_Salary     84998 non-null   float64
 9   Num_Bank_Accounts         100000 non-null  int64  
 10  Num_Credit_Card           100000 non-null  int64  
 11  Interest_Rate             100000 non-null  int64  
 12  Num_of_Loan               100000 non-null  object 
 13  Type_of_Loan              88592 non-null   object 
 14  Delay_from_due_date       100000 non-null  int64  
 15  Num_of_Delayed_Payment    92998 non-null   object 
 16  Changed_Credit_Limit      100000 non-null  object 
 17  Num_Credit_Inquiries      98035 non-null   float64
 18  Credit_Mix                100000 non-null  object 
 19  Outstanding_Debt          100000 non-null  object 
 20  Credit_Utilization_Ratio  100000 non-null  float64
 21  Credit_History_Age        90970 non-null   object 
 22  Payment_of_Min_Amount     100000 non-null  object 
 23  Total_EMI_per_month       100000 non-null  float64
 24  Amount_invested_monthly   95521 non-null   object 
 25  Payment_Behaviour         100000 non-null  object 
 26  Monthly_Balance           98800 non-null   object 
 27  Credit_Score              100000 non-null  object 
dtypes: float64(4), int64(4), object(20)
memory usage: 21.4+ MB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 27 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   ID                        50000 non-null  object 
 1   Customer_ID               50000 non-null  object 
 2   Month                     50000 non-null  object 
 3   Name                      44985 non-null  object 
 4   Age                       50000 non-null  object 
 5   SSN                       50000 non-null  object 
 6   Occupation                50000 non-null  object 
 7   Annual_Income             50000 non-null  object 
 8   Monthly_Inhand_Salary     42502 non-null  float64
 9   Num_Bank_Accounts         50000 non-null  int64  
 10  Num_Credit_Card           50000 non-null  int64  
 11  Interest_Rate             50000 non-null  int64  
 12  Num_of_Loan               50000 non-null  object 
 13  Type_of_Loan              44296 non-null  object 
 14  Delay_from_due_date       50000 non-null  int64  
 15  Num_of_Delayed_Payment    46502 non-null  object 
 16  Changed_Credit_Limit      50000 non-null  object 
 17  Num_Credit_Inquiries      48965 non-null  float64
 18  Credit_Mix                50000 non-null  object 
 19  Outstanding_Debt          50000 non-null  object 
 20  Credit_Utilization_Ratio  50000 non-null  float64
 21  Credit_History_Age        45530 non-null  object 
 22  Payment_of_Min_Amount     50000 non-null  object 
 23  Total_EMI_per_month       50000 non-null  float64
 24  Amount_invested_monthly   47729 non-null  object 
 25  Payment_Behaviour         50000 non-null  object 
 26  Monthly_Balance           49438 non-null  object 
dtypes: float64(4), int64(4), object(19)
memory usage: 10.3+ MB
```



Data Cleaning Procedure

Dataset includes various financial parameters relevant to credit scoring

Packages: Numpy, Pandas, Spicy.stats, Regular expressions

- Checked the head of data set and the overall data structure
- Checked the shape and data types of features to understand dataset dimensions and feature characteristics
- Fixed data types to numeric in Pandas: value_counts(), astype(), infer_objects(), convert_dtypes()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150000 entries, 0 to 149999
Data columns (total 32 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   ID                        150000 non-null  int64  
 1   Customer_ID               150000 non-null  int64  
 2   Month                     150000 non-null  int64  
 3   Name                      149988 non-null  object 
 4   Age                       150000 non-null  int64  
 5   SSN                       141600 non-null  float64
 6   Occupation                149994 non-null  object 
 7   Annual_Income             150000 non-null  float64
 8   Monthly_Inhand_Salary     127500 non-null  float64
 9   Num_Bank_Accounts         150000 non-null  int64  
 10  Num_Credit_Card           150000 non-null  int64  
 11  Interest_Rate             150000 non-null  int64  
 12  Num_of_Loan               150000 non-null  int64  
 13  Type_of_Loan              150000 non-null  object 
 14  Delay_from_due_date       150000 non-null  int64  
 15  Num_of_Delayed_Payment    139500 non-null  float64
 16  Changed_Credit_Limit      146850 non-null  float64
 17  Num_Credit_Inquiries      147000 non-null  float64
 18  Credit_Mix                149092 non-null  object 
 19  Outstanding_Debt          150000 non-null  float64
 20  Credit_Utilization_Ratio  150000 non-null  float64
 21  Credit_History_Age        136500 non-null  float64
 22  Payment_of_Min_Amount     150000 non-null  object 
 23  Total_EMI_per_month       150000 non-null  float64
 24  Amount_invested_monthly   143250 non-null  float64
 25  Payment_Behaviour         149196 non-null  object 
 26  Monthly_Balance           148238 non-null  float64
 27  Credit_Score              100000 non-null  object 
 28  Occupation_Num            150000 non-null  int8   
 29  Credit_Mix_Num            150000 non-null  int8   
 30  Payment_of_Min_Amount_Num 150000 non-null  int8   
 31  Payment_Behaviour_Num     150000 non-null  int8   
dtypes: float64(12), int64(9), int8(4), object(7)
memory usage: 32.6+ MB
```

Data Cleaning Procedure

- Assigned categorical types to numeric types
- For object columns, detected missing values, identifies NaN values in a given column and replaces them with the mode (most frequent value) within a specific group.
- For numeric columns, identified the minimum and maximum acceptable values for each group.
- Numeric values outside this range are considered outliers and are set to NaN.
- Then, these NaN values are filled with the mode of their respective groups, similar to the first method.
- Others include: transforming 'Month' to numerical month format; standardizing 'Credit_History_Age' to represent the age in months, and more

```
df['Occupation_Num'] = df.Occupation.astype('category').cat.codes
df['Credit_Mix_Num'] = df.Credit_Mix.astype('category').cat.codes
df['Payment_of_Min_Amount_Num'] = df.Payment_of_Min_Amount.astype('category').cat.codes
df['Payment_Behaviour_Num'] = df.Payment_Behaviour.astype('category').cat.codes
df
```

Credit_Score	Occupation_Num	Credit_Mix_Num	Payment_of_Min_Amount_Num	Payment_Behaviour_Num
Good	12	-1	1	2
Good	12	1	1	3
Good	12	1	1	4
Good	12	1	1	5
Good	12	1	1	1
...	...	...	...	...
NaN	1	-1	2	5
NaN	9	1	0	5
NaN	9	1	1	3
NaN	9	1	1	2
NaN	9	-1	1	4

03

Visualization

Data Exploration

Exploratory Data Analysis (EDA) (1)

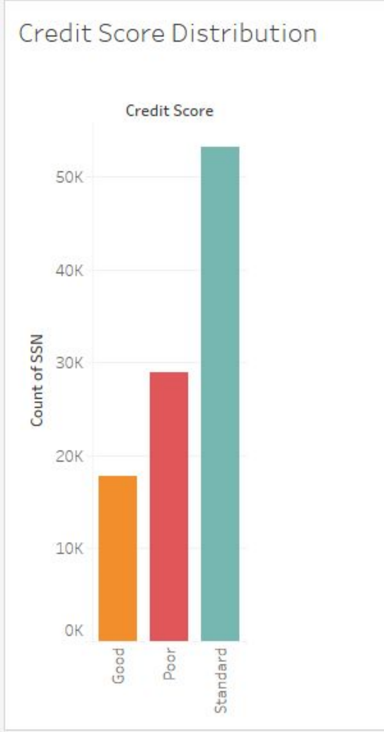
Credit Score

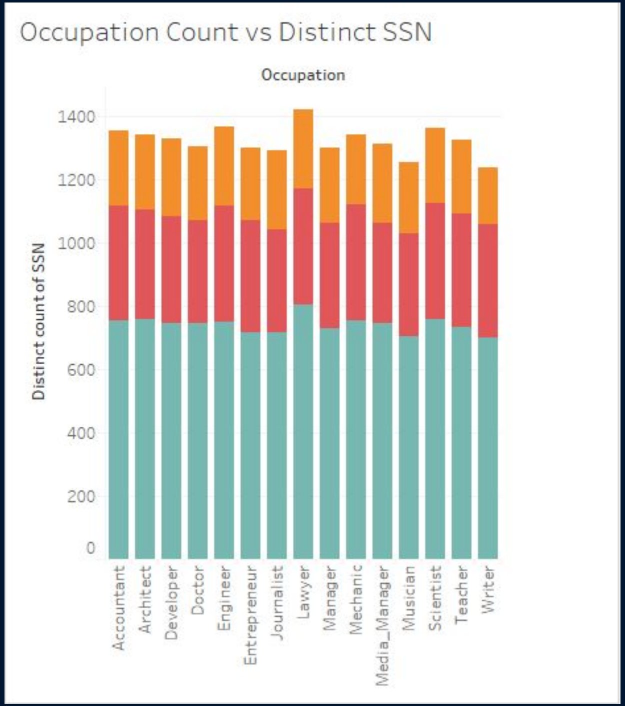
- Standard
- Good
- Poor

Payment of Minimum Amount

Credit Score	Payment of Min Amount: NM	No	Yes
Good	1,661	2,744	742
Poor	2,557	1,084	4,099
Standard	4,699	4,308	6,528

Exploratory Data Analysis (EDA) (4)

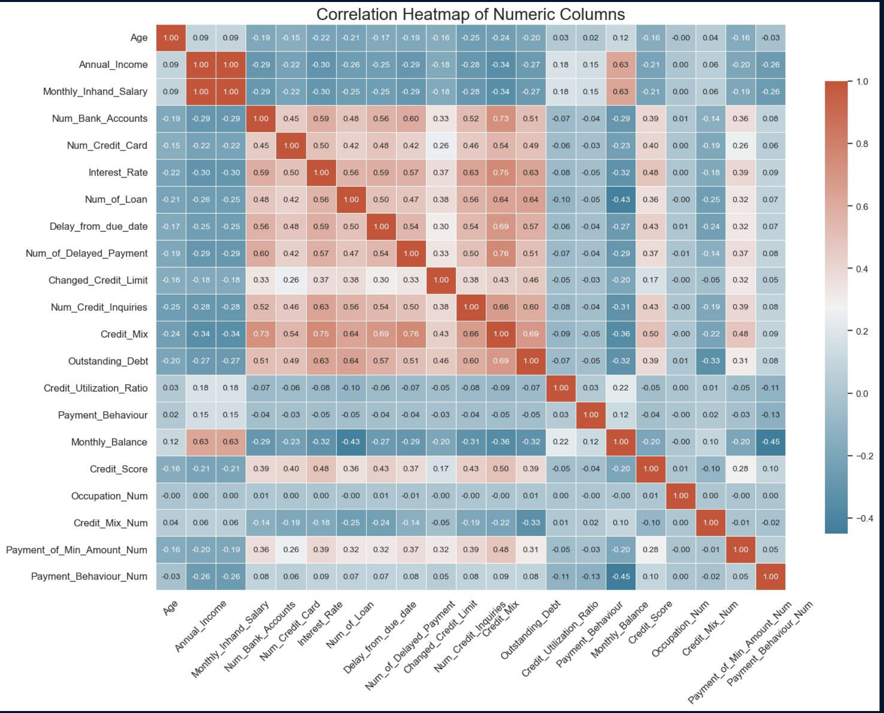




Credit Score

- Standard
- Good
- Poor

Heat Map



Color Scale: The colors represent the strength and direction of the correlation between variables. The scale goes from -1 to 1:

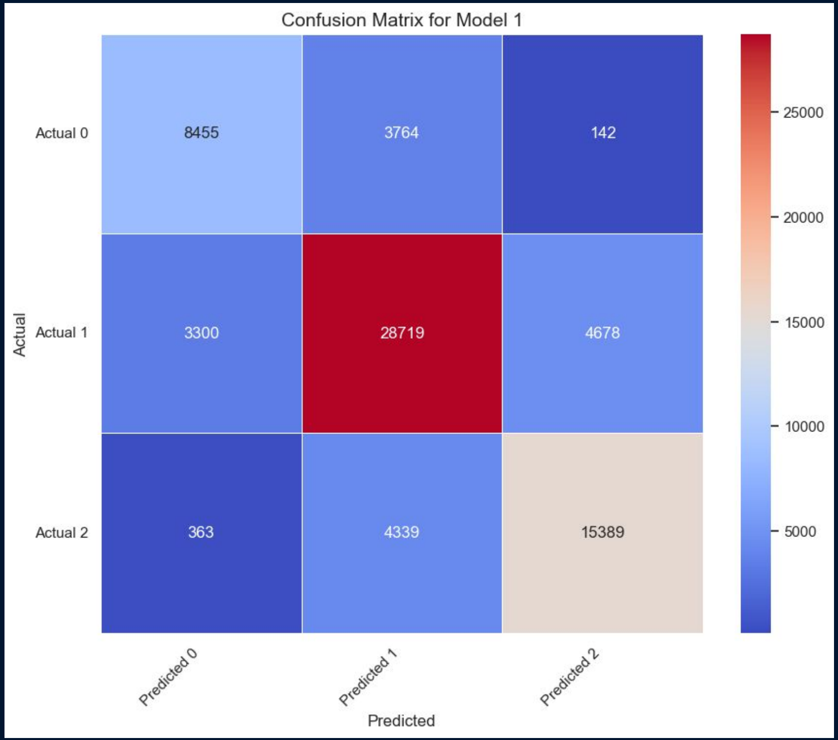
- 1 (warmer colors): A perfect positive correlation, meaning that as one variable increases, the other variable also increases.
- 0 (neutral colors): No linear correlation between the variables.
- -1 (cooler color): A perfect negative correlation

Help with feature selection

04 Frontend

Integrated Machine Learning,
Decision Tree

First Model




```
Classification Report for Model 1:
              precision    recall  f1-score   support

           0       0.70      0.68      0.69     12361
           1       0.78      0.78      0.78     36697
           2       0.76      0.77      0.76     20091

    accuracy                           0.76     69149
   macro avg       0.75      0.74      0.75     69149
weighted avg       0.76      0.76      0.76     69149


Accuracy Score for Model 1: 76.01%
```

Feature Selection:

```
X  = Num_Bank_Accounts, Num_Credit_Card, Interest_Rate,
Delay_from_due_date, Num_Credit_Inquiries,
Credit_Mix, Outstanding_Debt
Y  = Credit_Score
```

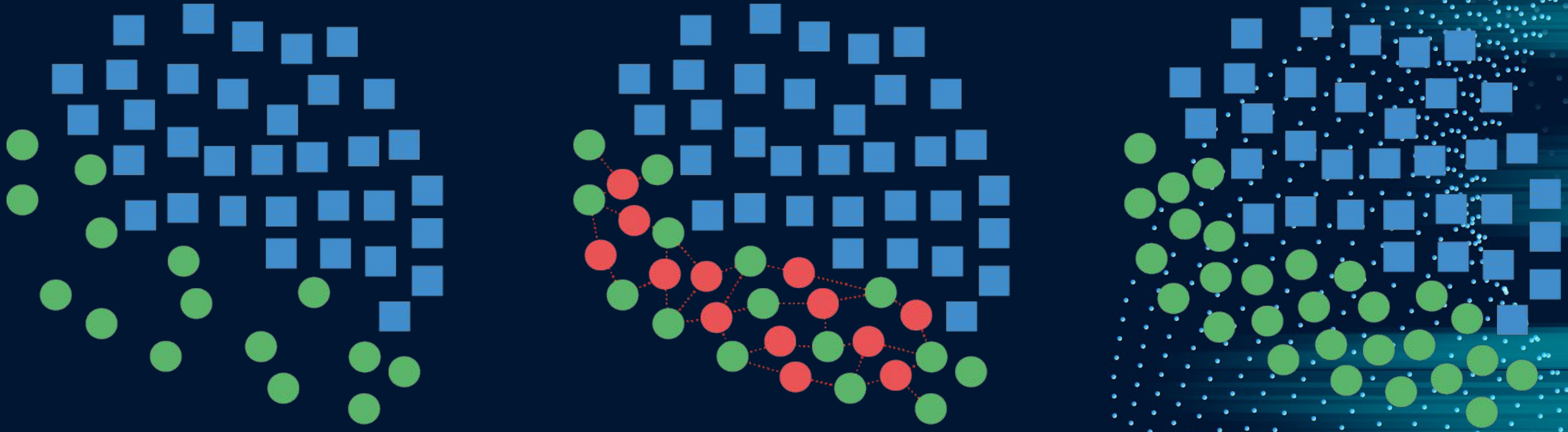
Modeling Improvements

Utilized DecisionTreeClassifier and RandomForestClassifier for the modeling techniques, which build a tree or map of the significant variables and determine feature importances in the case of RandomForestClassifier.

	Variable	Value
12	Outstanding_Debt	0.168829
5	Interest_Rate	0.094328
7	Delay_from_due_date	0.088565
9	Changed_Credit_Limit	0.069864
11	Credit_Mix	0.057873
15	Monthly_Balance	0.057710
13	Credit_Utilization_Ratio	0.056466
8	Num_of_Delayed_Payment	0.055630
2	Monthly_Inhand_Salary	0.053600
1	Annual_Income	0.052900
10	Num_Credit_Inquiries	0.050391
0	Age	0.047743
3	Num_Bank_Accounts	0.046424
4	Num_Credit_Card	0.042566
6	Num_of_Loan	0.032386
14	Payment_Behaviour	0.024725

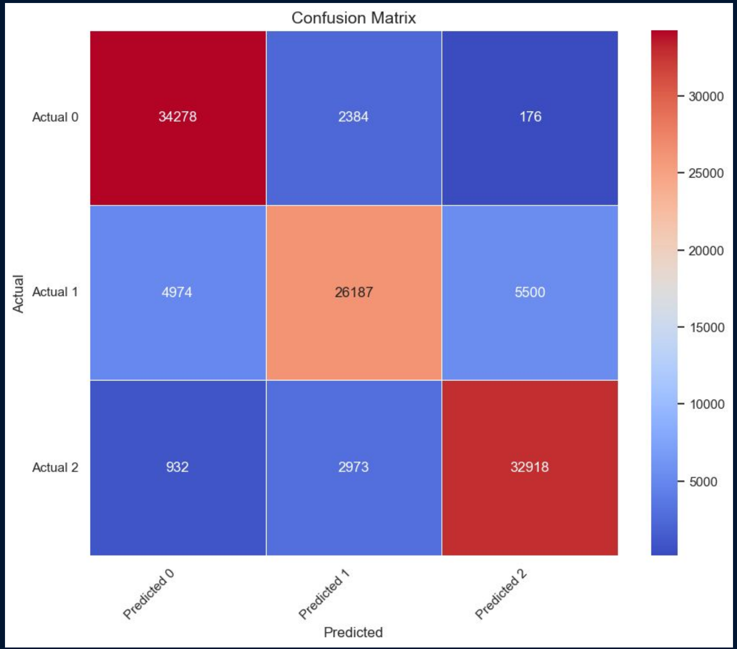
Synthetic Minority Over-sampling Technique

Utilized RandomOverSampler to oversample minority class to ensure balance among the classes by generating synthetic samples



Highest Accuracy on SMOTE

Highest Accuracy achieved on SMOTE Model
followed by RandomForestClassifier and then by DecisionTreeClassifier.




```
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.93      0.91      5254
           1       0.90      0.81      0.85      5254
           2       0.82      0.92      0.87      2868

    accuracy                           0.88     13376
   macro avg       0.87      0.88      0.88     13376
weighted avg       0.88      0.88      0.88     13376


Accuracy Score: 87.87%
```

05

Summary and Impact

Accuracy and Risk Profile

Risk Profile

- Our latest model perform well, with high precision, recall, and F1-scores.
- With an accuracy score of 88%, our model had overall good performance with the dataset.
- Prospective businesses will need to assess of this model is within risk tolerances

0 - Good
1 - Standard
2 - Poor

Classification Report:

	precision	recall	f1-score	support
0	0.90	0.93	0.91	5254
1	0.90	0.81	0.85	5254
2	0.82	0.92	0.87	2868
accuracy			0.88	13376
macro avg	0.87	0.88	0.88	13376
weighted avg	0.88	0.88	0.88	13376

Accuracy Score: 87.87%

Strengths, Weaknesses and Impact for Stakeholders

Model Strengths:

- The model demonstrates strong predictive capabilities for identifying good and poor credit scores, suggesting that it can be trusted to flag individuals at either end of the credit spectrum effectively.

Model Weaknesses

- The model has room for improvement in correctly classifying individuals with standard credit scores. Strategies to improve recall in this category could enhance overall model performance.
- The model shows a tendency to incorrectly classify actual standard and poor credit scores as good (as seen by the non-diagonal cells in the confusion matrix). Focusing on reducing these types of misclassifications could be beneficial.

Potential Impact

- The model could be used as a screening tool to identify individuals who may need credit counseling or financial advice, as it effectively identifies poor credit scores.

Potential for the Future

- Further tuning of the model's hyperparameters could potentially improve accuracy, especially for the standard credit score class.
- Additional feature engineering or the inclusion of more relevant data could provide the model with better discriminative power.
- Exploring other algorithms or ensemble methods might yield better recall for standard credit scores without compromising precision for other classes.