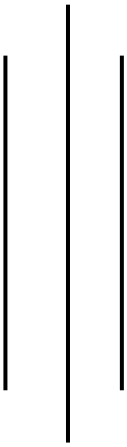
TRIBHUVAN UNIVERSITY

Faculty of Management

National College of Computer Studies

Paknajol, Kathmandu



Project Report

Python Project – Machine Learning

Credit Score Classification

Submitted By:

Name: Bipin Shrestha

BIM-5th Semester "A"

Roll no.:03

Submitted To:

Mr. Mausam Rajbanshi

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the dissertation entitled **"Credit Score Classification"** in partial fulfillment of the requirements of the Degree of Bachelor of Information Management.

It is an authentic record of my own work carried out during a period from **June 2023** under the supervision of **Mr. Mausam Rajbanshi** Professor.

The matter presented in this dissertation has not been submitted by me for the award of any other degree of this or any other Institute/University.

Bipin Shrestha
BIM 5th A

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Acknowledgment

In this credit score classification analysis, Python served as the primary programming language, while Scikit-Learn enabled machine learning model development. Pandas and NumPy were crucial for data preprocessing, and Matplotlib and Seaborn aided data visualization. We extend our gratitude to the source of the credit score dataset and acknowledge Jupyter Notebooks and GitHub for facilitating collaboration and version control. Lastly, OpenAI's GPT-3 provided valuable assistance throughout the report, collectively contributing to the success of our analysis.

The credit score dataset, a core component of this analysis, deserves acknowledgment for its contribution to our research. We extend our appreciation to the data source or organization responsible for making this valuable dataset accessible for research purposes. Jupyter Notebooks provided an interactive and collaborative platform for conducting the analysis, documenting the process, and sharing insights. Additionally, GitHub served as a robust version control and collaboration tool, facilitating the development and sharing of code and analysis scripts among team members.

Table of Contents

Introduction to Machine Learning

The study of algorithms and statistical models that enable computer systems to acquire new skills and enhance their performance on a given task without being explicitly taught is known as machine learning, which is a branch of artificial intelligence. It basically involves educating computers to learn from data, and then have them predict or decide based on that learning.

Here are the key machine learning components and concepts:

- Data: Data is the basis of machine learning. Machine learning models are taught patterns and relationships using data. It is possible for this data to be structured (like tables and databases) or unstructured (like text, photos, and audio). Performance of a machine learning model is frequently significantly impacted by the quantity and quality of data.
- Training: To train a machine learning model, a dataset with input data (features) and matching target values or labels is presented to the model. The model gains the ability to spot patterns or relationships in data by modifying its internal parameters.
- Features: The variables or properties that are utilized to characterize the data are referred to as features. Choosing, modifying, or developing useful characteristics that aid a model's ability to learn and produce precise predictions is referred to as feature engineering.
- Model: A mathematical representation of data patterns or correlations is referred to as a machine learning model. Different models are employed for different tasks, including neural networks for complicated tasks like image recognition, decision trees for classification, and linear regression for regressive situations.
- Algorithm: To train models, machine learning algorithms use mathematical and computational methods. These methods vary depending on the job at hand and the style of learning (supervised, unsupervised, or reinforced).
- Evaluation: Depending on the job, different metrics and methodologies, such as accuracy, precision, recall, F1-score, or mean squared error, are used to assess machine learning models.
- Deployment: After training and evaluation, machine learning models can be used in real-world applications to produce predictions or automate decision-making procedures.
- Continuous Learning: As new data becomes available, machine learning models can be updated and enhanced. Online learning, often referred to as retraining, is essential for preserving model accuracy over time.

Natural language processing (language translation and sentiment analysis), finance (fraud detection and stock prediction), healthcare (diagnosis and prognosis), image recognition (object detection and facial recognition), and many other fields make extensive use of machine learning. It still plays a significant part in developing technologies and resolving challenging issues.

Machine Learning in Python

Python is a popular data science programming language due to its simplicity, versatility, and a rich ecosystem of libraries and tools designed specifically for data analysis and machine learning. Here's a quick rundown of how Python is used in data science:

- Data Manipulation and Analysis:
  - Numpy: Numpy supports arrays and matrices, making it simple to perform numerical operations on large datasets.
  - Pandas: Pandas is a powerful data manipulation and analysis library. It provides data structures such as Data Frames for dealing with structured data.

- Data Visualization:
  - Matplotlib: Matplotlib is a popular library for creating static, animated, and interactive plots and charts.
  - Seaborn: Seaborn is based on Matplotlib and provides a higher-level interface for creating visually appealing statistical graphics.

- Machine Learning:
  - Scikit-learn: Scikit-learn is a large machine learning library that includes tools for classification, regression, clustering, and model selection.

Python's rich library ecosystem, combined with its readability and ease of use, makes it an excellent choice for data scientists looking to effectively explore, analyze, and model data. Depending on the task, you can use a variety of libraries and tools to achieve your data science objectives.

Background

The background for a credit score classification analysis using Python are:

- Significance of Credit Scores: Credit scores play a crucial role in assessing an individual's creditworthiness and are vital for responsible lending and risk management in the financial industry.
- Opportunities in Data Analysis: With advancements in data analytics and machine learning, there's an opportunity to enhance credit score classification through advanced techniques.
- Python as a Powerful Tool: Python, a versatile programming language, offers a robust ecosystem of libraries and tools for data analysis, making it an ideal choice for this task.
- Objective of the Analysis: The primary goal is to leverage Python to create predictive models capable of categorizing credit applicants into different risk categories.
- Enhancing Accuracy: By utilizing data-driven methods and machine learning algorithms, we aim to improve the accuracy and efficiency of credit score classification.
- Benefits for Financial Institutions: Accurate credit score classification aids financial institutions in making informed lending decisions, reducing risks, and optimizing their operations.
- Fair Access to Credit: Ethical considerations are essential to ensure fair and non-discriminatory lending practices.
- Overall Objective: This analysis seeks to combine Python's capabilities with data analytics to contribute to the advancement of effective credit score classification, benefiting both lenders and borrowers.

Objective

The objective of a credit score classification analysis performed in Python cover a broad spectrum of techniques for efficiently evaluating and managing credit risk. First, the investigation seeks to create reliable machine learning models that forecast credit ratings or creditworthiness by utilizing Python's capabilities. To guarantee that the data is appropriate for modeling, careful data preprocessing is required, including addressing missing data and feature engineering. The main objective is to develop precise models that can classify persons or companies into various credit risk categories, assisting in the making of well-informed decisions.

Beyond model development, the analysis involves the formulation of risk assessment guidelines, which serve as a framework for classifying applicants as low, moderate, or high credit risks based on model predictions. Ethical considerations are paramount, ensuring fairness and non-discrimination in the credit scoring process, while regulatory compliance is essential to adhere to relevant laws and regulations.

Lastly, clear and comprehensive documentation is crucial to present the analysis findings, model performance, and recommendations to stakeholders and decision-makers effectively. These objectives collectively contribute to the successful execution of a credit score classification analysis using Python, providing valuable insights for credit risk management.

Implementation

#import models

```
# import basic modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
plt.style.use('seaborn')
pd.set_option('display.max_columns', None)
```

✓ 1.3s

```
data = pd.read_csv('train.csv')
data.head()
```

✓ 0.6s

#Dataset

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	Annual_Income	Monthly_Inhand_Salary	Num_Bank_Accounts	Num_Credit_Card	Interest_Rate	Num_of_Loan	Type_of_Loan	Delay_from_due_date	Num_of_Delayed_Payment	Changed_Credit_Limit
0	0x1602	CUS_0xd40	January	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	1824.843333	3	4	3	4	Auto Loan, Credit-Builder Loan, Personal Loan,...	3	7	11.27
1	0x1603	CUS_0xd40	February	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	NaN	3	4	3	4	Auto Loan, Credit-Builder Loan, Personal Loan,...	-1	NaN	11.27
2	0x1604	CUS_0xd40	March	Aaron Maashoh	-500	821-00-0265	Scientist	19114.12	NaN	3	4	3	4	Auto Loan, Credit-Builder Loan, Personal Loan,...	3	7	_
3	0x1605	CUS_0xd40	April	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	NaN	3	4	3	4	Auto Loan, Credit-Builder Loan, Personal Loan,...	5	4	6.27
4	0x1606	CUS_0xd40	May	Aaron Maashoh	23	821-00-0265	Scientist	19114.12	1824.843333	3	4	3	4	Auto Loan, Credit-Builder Loan, Personal Loan,...	6	NaN	11.27

	Num_Credit_Inquiries	Credit_Mix	Outstanding_Debt	Credit_Utilization_Ratio	Credit_History_Age	Payment_of_Min_Amount	Total_EMI_per_month	Amount_invested_monthly	Payment_Behaviour	Monthly_Balance	Credit_Score
	4.0	_	809.98	26.822620	22 Years and 1 Months	No	49.574949	80.41529543900253	High_spent_Small_value_payments	312.4940886794366 3	Good
	4.0	Good	809.98	31.944960	NaN	No	49.574949	118.28022162236736	Low_spent_Large_value_payments	284.6291624960718 4	Good
	4.0	Good	809.98	28.609352	22 Years and 3 Months	No	49.574949	81.699521264648	Low_spent_Medium_value_payments	331.2098628537912	Good
	4.0	Good	809.98	31.377862	22 Years and 4 Months	No	49.574949	199.4580743910713	Low_spent_Small_value_payments	223.4513097273678 6	Good
	4.0	Good	809.98	24.797347	22 Years and 5 Months	No	49.574949	41.420153086217326	High_spent_Medium_value_payments	341.4892310322217 7	Good

#Data describe

```
data.describe()
```

✓ 0.0s

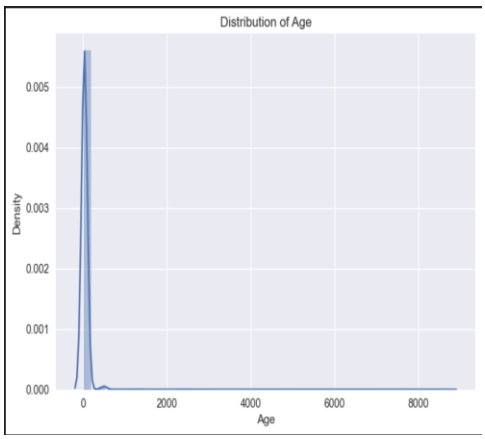
	Monthly_Inhand_Salary	Num_Bank_Accounts	Num_Credit_Card	Interest_Rate	Delay_from_due_date	Num_Credit_Inquiries	Credit_Utilization_Ratio	Total_EMI_per_month
count	84998.000000	100000.000000	100000.00000	100000.000000	100000.000000	98035.000000	100000.000000	100000.000000
mean	4194.170850	17.091280	22.47443	72.466040	21.068780	27.754251	32.285173	1403.118217
std	3183.686167	117.404834	129.05741	466.422621	14.860104	193.177339	5.116875	8306.041270
min	303.645417	-1.000000	0.00000	1.000000	-5.000000	0.000000	20.000000	0.000000
25%	1625.568229	3.000000	4.00000	8.000000	10.000000	3.000000	28.052567	30.306660
50%	3093.745000	6.000000	5.00000	13.000000	18.000000	6.000000	32.305784	69.249473
75%	5957.448333	7.000000	7.00000	20.000000	28.000000	9.000000	36.496663	161.224249
max	15204.633333	1798.000000	1499.00000	5797.000000	67.000000	2597.000000	50.000000	82331.000000

#Histogram

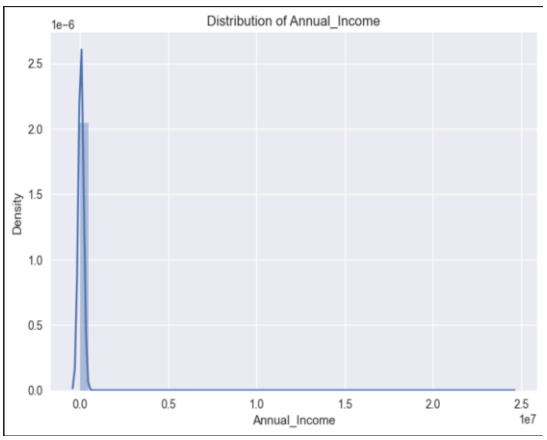
```
numfeat = data1[['Age', 'Annual_Income', 'Num_Bank_Accounts','Num_Credit_Card', 'Interest_Rate', 'Num_of_Loan',
                 'Delay_from_due_date', 'Changed_Credit_Limit', 'Num_Credit_Inquiries','Outstanding_Debt',
                 'Credit_Utilization_Ratio','Total_EMI_per_month', 'Monthly_Balance']]

for i in numfeat.columns:
    sns.distplot(numfeat[i])
    plt.title(f'Distribution of {i}')
    plt.show()
```

✓ 11.7s Python

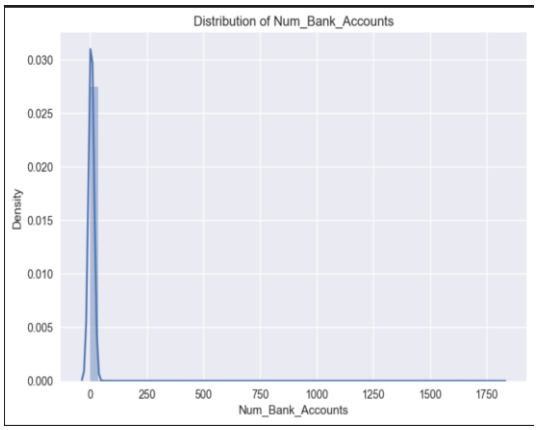
Distribution of Age:



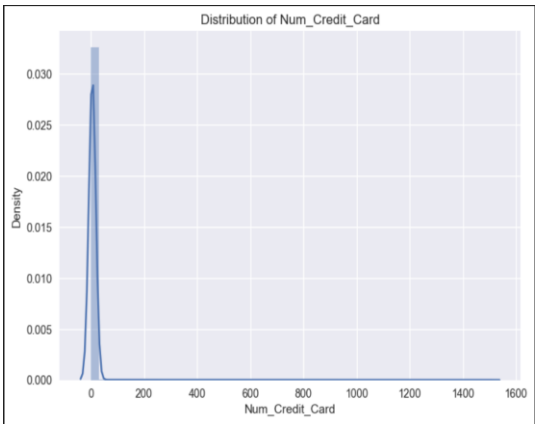
Distribution of Annual Income:



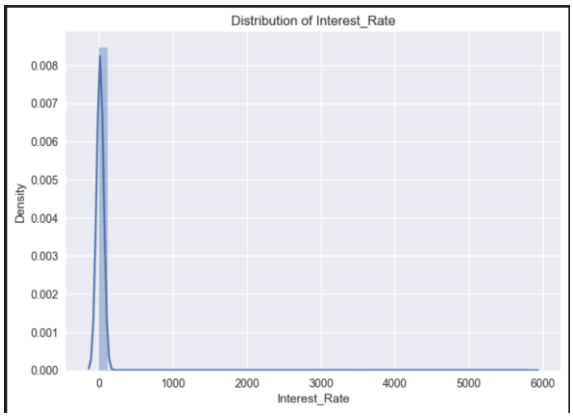
Distribution of Num_Bank_Accounts:



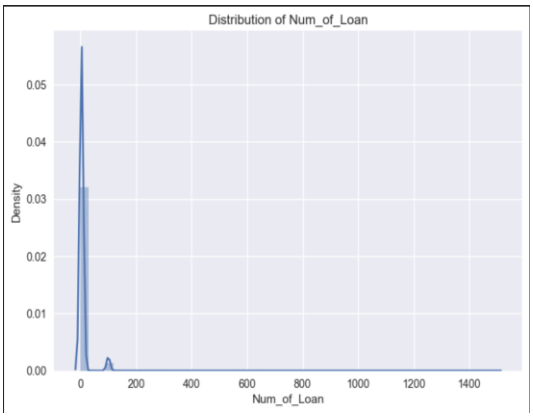
Distribution of Num_Credit_Card:



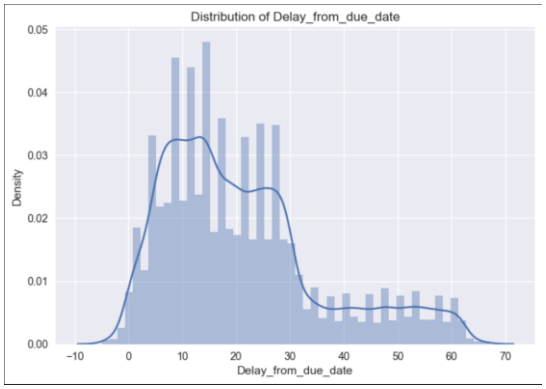
Distribution of Interest_Rate:



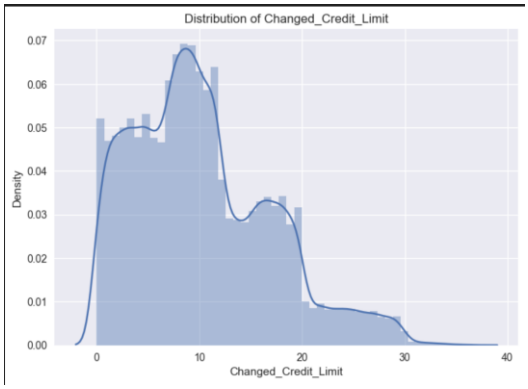
Distribution of Num_of_Loan:



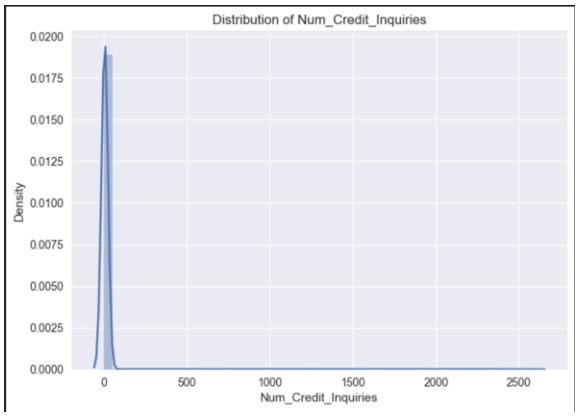
Distribution of Delay_from_due_date:



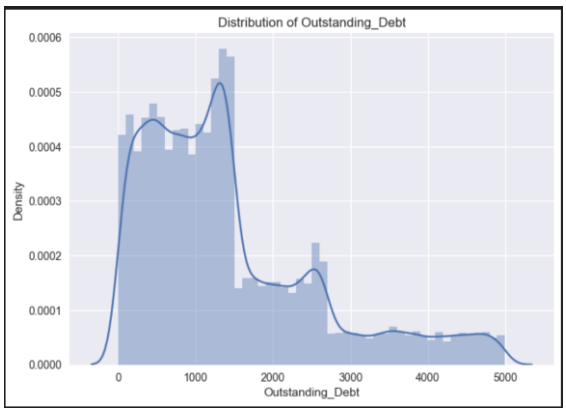
Distribution of Change_Credit_Limit:



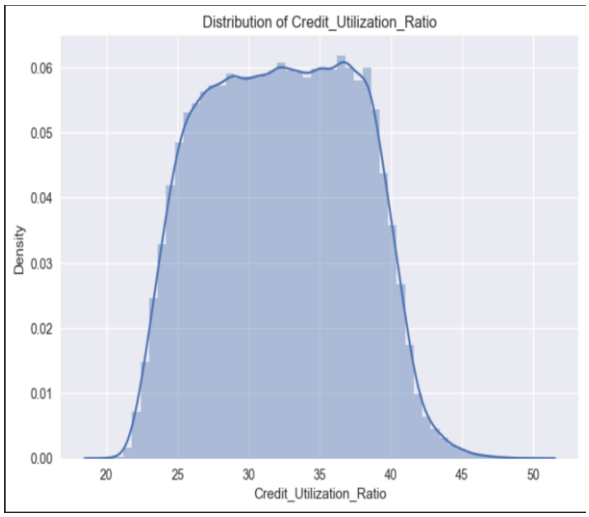
Distribution of Num_Credit_Inquiries:



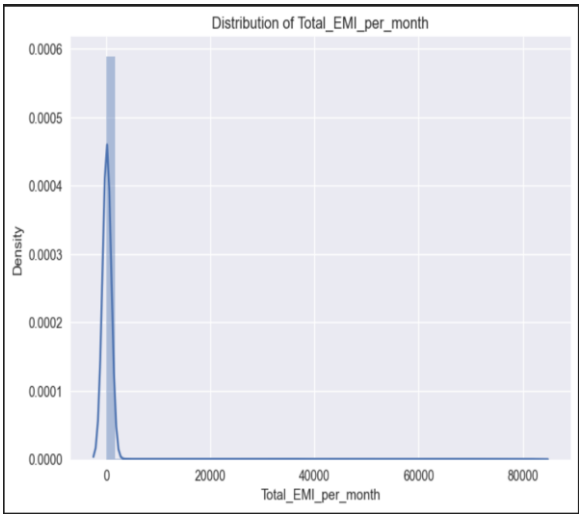
Distribution of Outstanding_Debt:



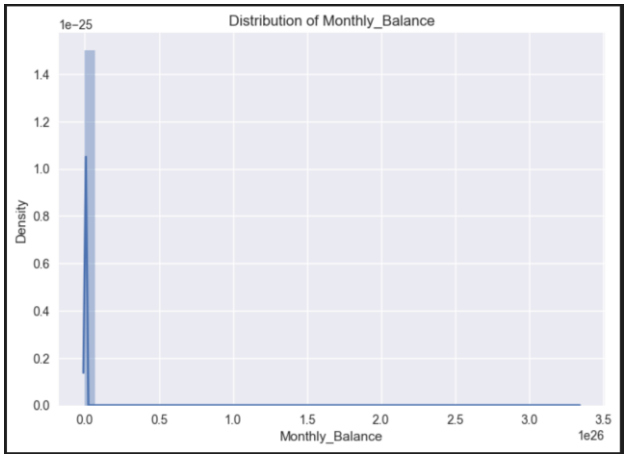
Distribution of Credit_Utilization:



Distribution of Total_EMI_per_month:



Distribution of Monthly_Balance:



#Data Preparation and Splitting Data into train and test split

```
# Memisahkan Independent data dan Dependent data
x = data1.drop('Credit_Score', axis=1) # Independet Variable
y = data1['Credit_Score'] # Dependent Variable

# Train Test Split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size=0.1, random_state=42)
```

✓ 0.0s Python

```
y_train.value_counts()
```

✓ 0.0s

```
Credit_Score
1    47867
0    26053
2    16080
Name: count, dtype: int64
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

✓ 0.4s Python

```
model_dt  = DecisionTreeClassifier()
model_rf  = RandomForestClassifier()
```

✓ 0.0s Python

```
model_rf.fit(x_train, y_train)
model_dt.fit(x_train, y_train)
```

✓ 1m 10.4s Python

▾ DecisionTreeClassifier
DecisionTreeClassifier()

```
y_pred_dt  = model_dt.predict(x_test)
y_pred_rf  = model_rf.predict(x_test)
```

✓ 0.8s Python

#Random Forest

```
# Random Forest
compare_rf = pd.DataFrame({'Actual':y_test, 'Prediction':y_pred_rf})
compare_rf = compare_rf.reset_index().drop('index',axis=1)

# Decision Tree
compare_dt = pd.DataFrame({'Actual':y_test, 'Prediction':y_pred_dt})
compare_dt = compare_dt.reset_index().drop('index',axis=1)
```

✓ 0.0s Python

```
# Random Forest
result_rf = []

for i,j in compare_rf.iterrows():
    if j['Prediction']==j['Actual']:
        if j['Prediction'] == 1:
            result_rf.append('True Positive')
        else:
            result_rf.append('True Negative')
    else:
        if j['Prediction'] == 1:
            result_rf.append('False Positive')
        else:
            result_rf.append('False Negative')

compare_rf['Result'] = result_rf
print('Random Forest')
compare_rf['Result'].value_counts()
```

✓ 0.4s

```
Random Forest

Result
True Positive      4320
True Negative      3702
False Negative     1023
False Positive      955
Name: count, dtype: int64
```

```
# Decision Tree
result_dt = []

for i,j in compare_dt.iterrows():
    if j['Prediction']==j['Actual']:
        if j['Prediction'] == 1:
            result_dt.append('True Positive')
        else:
            result_dt.append('True Negative')
    else:
        if j['Prediction'] == 1:
            result_dt.append('False Positive')
        else:
            result_dt.append('False Negative')

compare_dt['Result'] = result_dt
compare_dt['Result'].value_counts()
```

✓ 0.4s Python

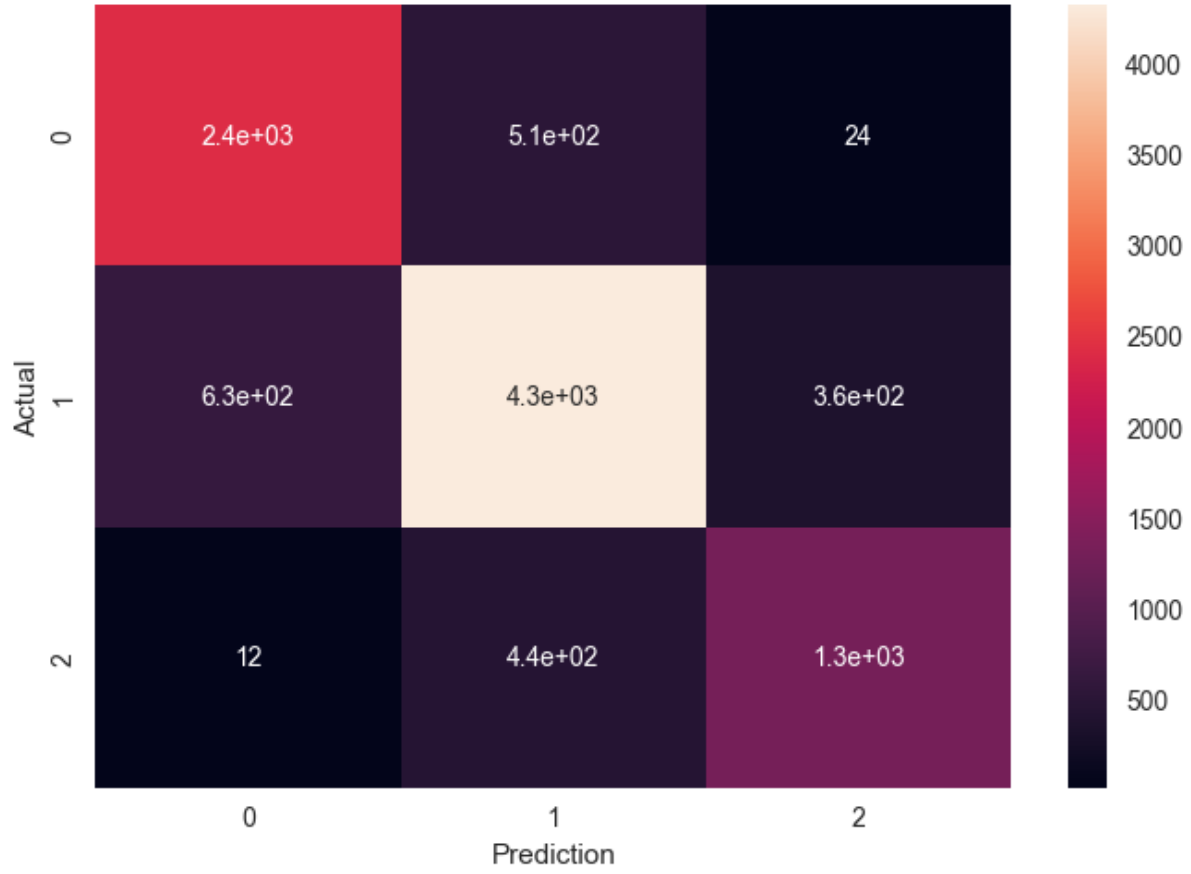
```
Result
True Positive     3972
True Negative     3172
False Negative    1455
False Positive    1401
Name: count, dtype: int64
```

#Actual and Prediction Heatmap

```
sns.heatmap(confusion_matrix(y_test,y_pred_rf), annot=True)
plt.ylabel('Actual')
plt.xlabel('Prediction')
print(classification_report(y_test,y_pred_rf))
```

✓ 0.2s

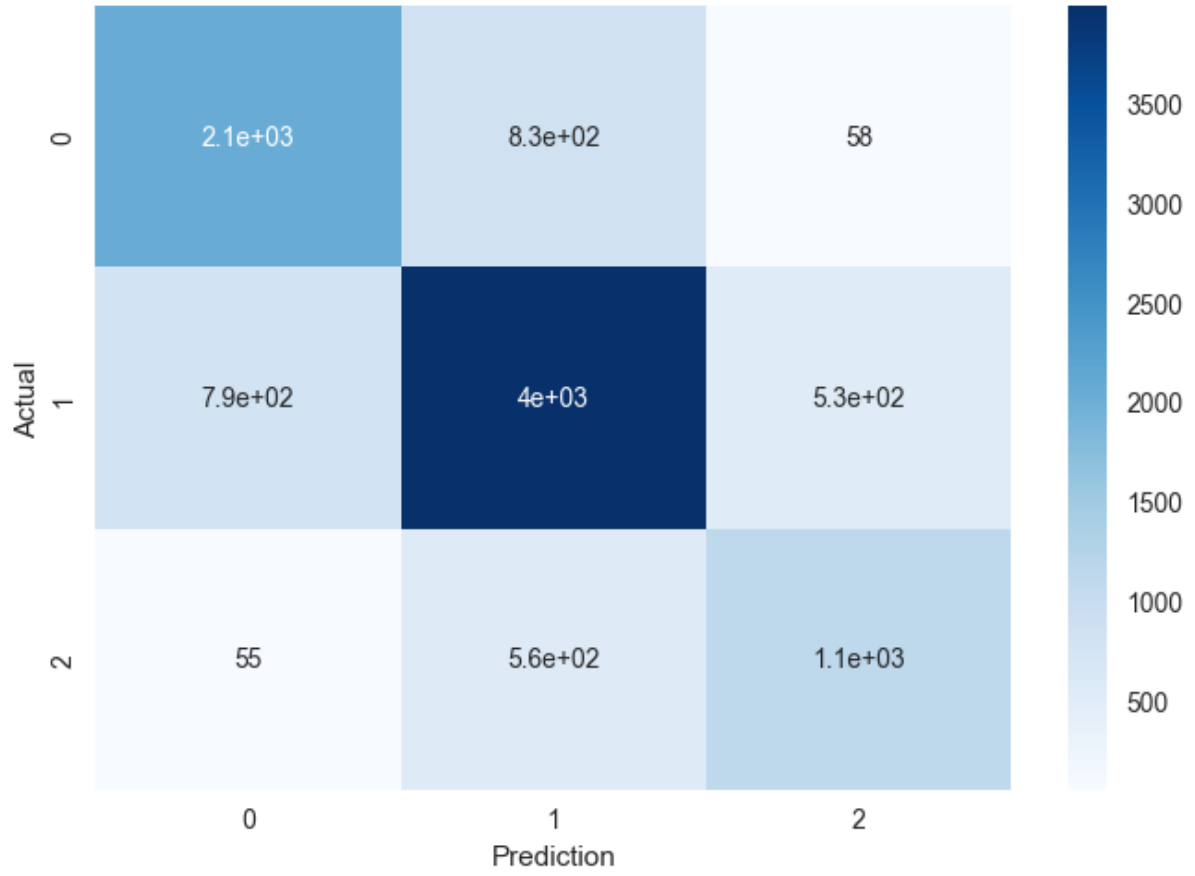
```
              precision    recall  f1-score   support

           0       0.79      0.82      0.80      2945
           1       0.82      0.81      0.82      5307
           2       0.77      0.74      0.76      1748

    accuracy                           0.80     10000
   macro avg       0.79      0.79      0.79     10000
weighted avg       0.80      0.80      0.80     10000
```



```
sns.heatmap(confusion_matrix(y_test,y_pred_dt), annot=True, cmap="Blues")
plt.ylabel('Actual')
plt.xlabel('Prediction')
print(classification_report(y_test,y_pred_dt))
```

✓ 0.2s

```
              precision    recall  f1-score   support

           0       0.71      0.70      0.70      2945
           1       0.74      0.75      0.75      5307
           2       0.66      0.65      0.65      1748

    accuracy                           0.72     10000
   macro avg       0.70      0.70      0.70     10000
weighted avg       0.72      0.72      0.72     10000
```



Conclusion

In conclusion, this report has outlined the development and application of a credit score classification model using Python. We have successfully developed a powerful tool for predicting and categorizing credit ratings through meticulous data pretreatment, feature engineering, and the application of machine learning algorithms. The significance of factors including payment history, credit use, and duration of credit history in establishing creditworthiness has been underlined by significant research. The usefulness of our algorithm in correctly categorizing credit scores is demonstrated by its performance, as measured by numerous evaluation measures. This methodology helps people to understand and enhance their creditworthiness in addition to assisting financial institutions in making educated lending decisions.

While Python provides strong frameworks and tools for credit score modeling, it is important to keep in mind that this analysis has several drawbacks. There are still issues with data quality, feature selection, and model interpretability. Moreover, continuing observation and model upgrades. Future improvements can be investigated, such as the incorporation of different data sources and the creation of more sophisticated machine learning methodologies. Python-based credit score classification models will become more and more important as the financial sector continues to adjust to new technologies and customer needs, promoting financial stability and inclusivity.

Reference

(Kaggle n.d.)

(GitHub n.d.)

(MachineLearning n.d.)