Credit Score Classification with Machine Learning

AMAN KHARWAL / DECEMBER 5, 2022 / MACHINE LEARNING

Banks and credit card companies calculate your credit score to determine your creditworthiness. It helps banks and credit card companies immediately to issue loans to customers with good creditworthiness. Today banks and credit card companies use Machine Learning **algorithms** to classify all the customers in their database based on their credit history. So, if you want to learn how to use Machine Learning for credit score classification, this article is for you. In this article, I will take you through the task of credit score classification with Machine Learning using **Python**.

Credit Score Classification

There are three credit scores that banks and credit card companies use to label their customers:

1. Good
2. Standard
3. Poor

A person with a good credit score will get loans from any bank and financial institution. For the task of Credit Score Classification, we need a labelled dataset with credit scores.

I found an ideal dataset for this task labelled according to the credit history of credit card customers. You can download the dataset **here**.

In the section below, I will take you through the task of credit score classification with Machine Learning using Python.

Credit Score Classification using Python

Let's start the task of credit score classification by importing the necessary Python libraries and the **dataset**:

```
import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
pio.templates.default = "plotly_white"

data = pd.read_csv("train.csv")
print(data.head())
```

```
     ID  Customer_ID  Month           Name   Age          SSN Occupation  \
0  5634         3392      1  Aaron Maashoh  23.0  821000265.0  Scientist
1  5635         3392      2  Aaron Maashoh  23.0  821000265.0  Scientist
2  5636         3392      3  Aaron Maashoh  23.0  821000265.0  Scientist
3  5637         3392      4  Aaron Maashoh  23.0  821000265.0  Scientist
4  5638         3392      5  Aaron Maashoh  23.0  821000265.0  Scientist
```

```
   Annual_Income  Monthly_Inhand_Salary  Num_Bank_Accounts  ...  Credit_Mix  \
0       19114.12            1824.843333                3.0  ...        Good
1       19114.12            1824.843333                3.0  ...        Good
2       19114.12            1824.843333                3.0  ...        Good
3       19114.12            1824.843333                3.0  ...        Good
4       19114.12            1824.843333                3.0  ...        Good

   Outstanding_Debt  Credit_Utilization_Ratio Credit_History_Age  \
0            809.98                 26.822620              265.0
1            809.98                 31.944960              266.0
2            809.98                 28.609352              267.0
3            809.98                 31.377862              268.0
4            809.98                 24.797347              269.0

   Payment_of_Min_Amount  Total_EMI_per_month  Amount_invested_monthly  \
0                     No            49.574949                 21.46538
1                     No            49.574949                 21.46538
2                     No            49.574949                 21.46538
3                     No            49.574949                 21.46538
4                     No            49.574949                 21.46538

                Payment_Behaviour  Monthly_Balance  Credit_Score
0   High_spent_Small_value_payments       312.494089          Good
1    Low_spent_Large_value_payments       284.629162          Good
2   Low_spent_Medium_value_payments       331.209863          Good
3    Low_spent_Small_value_payments       223.451310          Good
4  High_spent_Medium_value_payments       341.489231          Good

[5 rows x 28 columns]
```

Let’s have a look at the information about the columns in the dataset:

```
1 print(data.info())
```

```
<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  int64
 1   Customer_ID               100000 non-null  int64
 2   Month                     100000 non-null  int64
 3   Name                      100000 non-null  object
 4   Age                       100000 non-null  float64
 5   SSN                       100000 non-null  float64
 6   Occupation                100000 non-null  object
 7   Annual_Income             100000 non-null  float64
 8   Monthly_Inhand_Salary     100000 non-null  float64
 9   Num_Bank_Accounts         100000 non-null  float64
 10  Num_Credit_Card           100000 non-null  float64
 11  Interest_Rate             100000 non-null  float64
 12  Num_of_Loan               100000 non-null  float64
 13  Type_of_Loan              100000 non-null  object
 14  Delay_from_due_date       100000 non-null  float64
 15  Num_of_Delayed_Payment    100000 non-null  float64
 16  Changed_Credit_Limit      100000 non-null  float64
 17  Num_Credit_Inquiries      100000 non-null  float64
 18  Credit_Mix                100000 non-null  object
 19  Outstanding_Debt          100000 non-null  float64
 20  Credit_Utilization_Ratio  100000 non-null  float64
```

```
 21  Credit_History_Age        100000 non-null  float64
 22  Payment_of_Min_Amount     100000 non-null  object
 23  Total_EMI_per_month       100000 non-null  float64
 24  Amount_invested_monthly   100000 non-null  float64
 25  Payment_Behaviour         100000 non-null  object
 26  Monthly_Balance           100000 non-null  float64
 27  Credit_Score              100000 non-null  object
dtypes: float64(18), int64(3), object(7)
memory usage: 21.4+ MB
None
```

Before moving forward, let's have a look if the dataset has any null values or not:

```
1 print(data.isnull().sum())
```

```
ID                        0
Customer_ID               0
Month                     0
Name                      0
Age                       0
SSN                       0
Occupation                0
Annual_Income             0
Monthly_Inhand_Salary     0
Num_Bank_Accounts         0
Num_Credit_Card           0
Interest_Rate             0
Num_of_Loan               0
Type_of_Loan              0
Delay_from_due_date       0
Num_of_Delayed_Payment    0
Changed_Credit_Limit      0
Num_Credit_Inquiries      0
Credit_Mix                0
Outstanding_Debt          0
Credit_Utilization_Ratio  0
Credit_History_Age        0
Payment_of_Min_Amount     0
Total_EMI_per_month       0
Amount_invested_monthly   0
Payment_Behaviour         0
Monthly_Balance           0
Credit_Score              0
dtype: int64
```

The dataset doesn't have any null values. As this dataset is labelled, let's have a look at the Credit_Score column values:

```
data["Credit_Score"].value_counts()
```

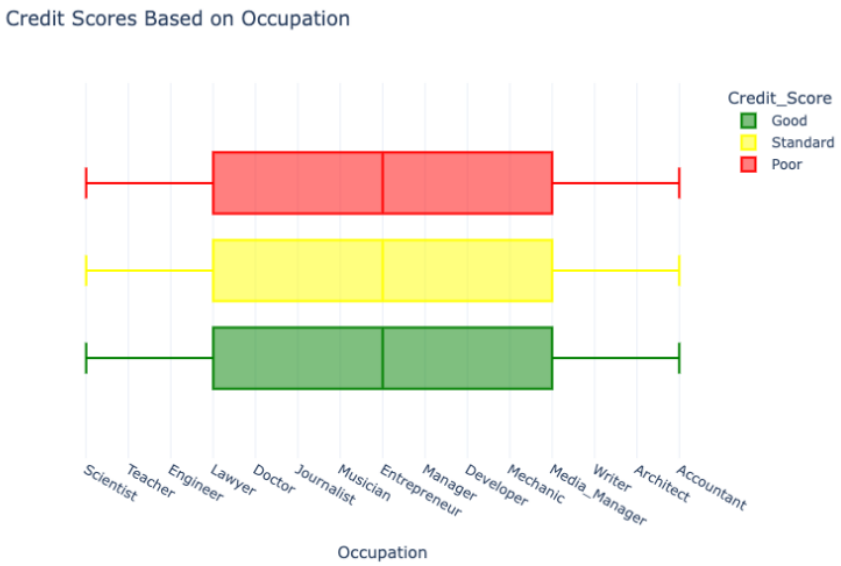
```
Standard    53174
Poor        28998
Good        17828
Name: Credit_Score, dtype: int64
```

Data Exploration

The dataset has many features that can train a Machine Learning model for credit score classification. Let's explore all the features one by one.

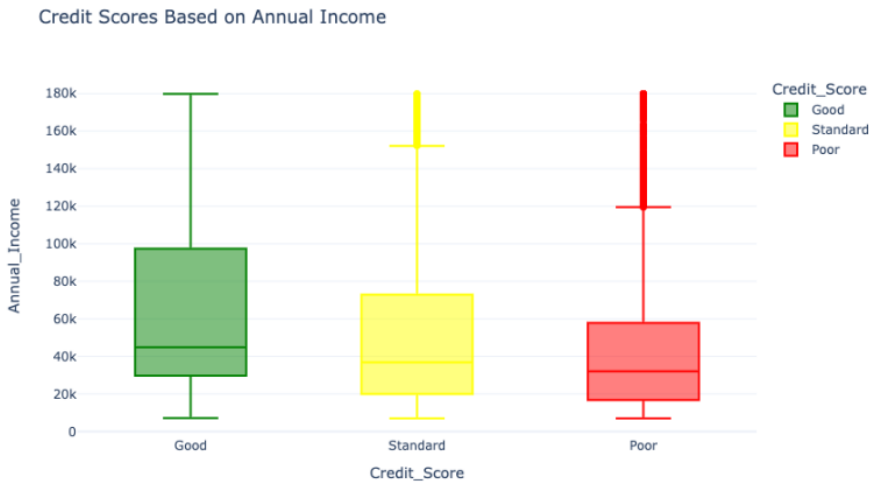
I will start by exploring the occupation feature to know if the occupation of the person affects credit scores:

```
fig = px.box(data,
             x="Occupation",
             color="Credit_Score",
             title="Credit Scores Based on Occupatio
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.show()
```



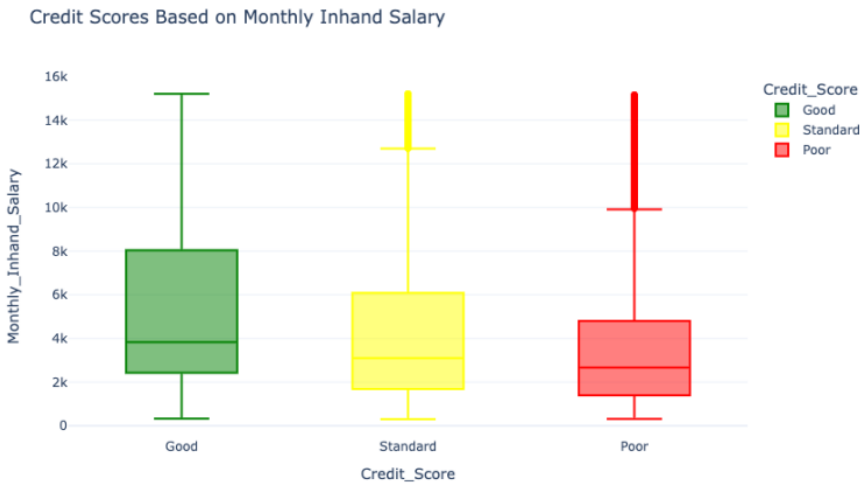
There's not much difference in the credit scores of all occupations mentioned in the data. Now let's explore whether the Annual Income of the person impacts your credit scores or not:

```
fig = px.box(data,
             x="Credit_Score",
             y="Annual_Income",
             color="Credit_Score",
             title="Credit Scores Based on Annual I
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



According to the above visualization, the more you earn annually, the better your credit score is. Now let's explore whether the monthly in-hand salary impacts credit scores or not:

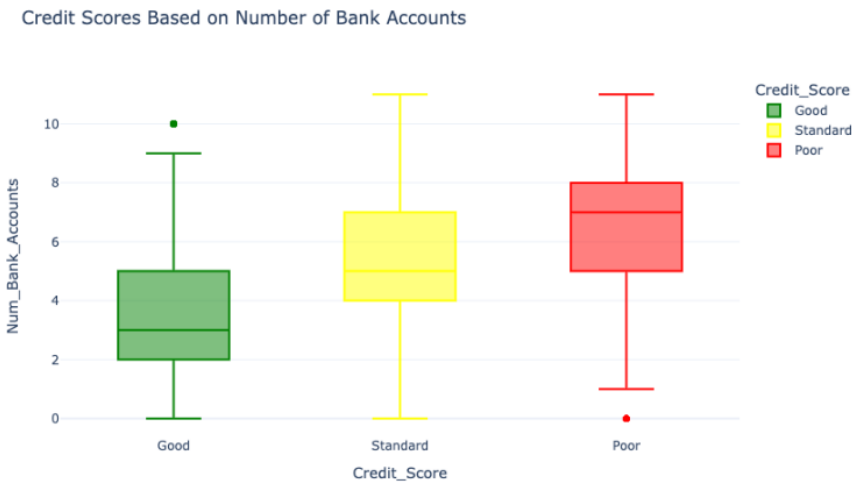

```
fig = px.box(data,
             x="Credit_Score",
             y="Monthly_Inhand_Salary",
             color="Credit_Score",
             title="Credit Scores Based on Monthly
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



Like annual income, the more monthly in-hand salary you earn, the better your credit score will become. Now let's see if having more bank accounts impacts credit scores or not:

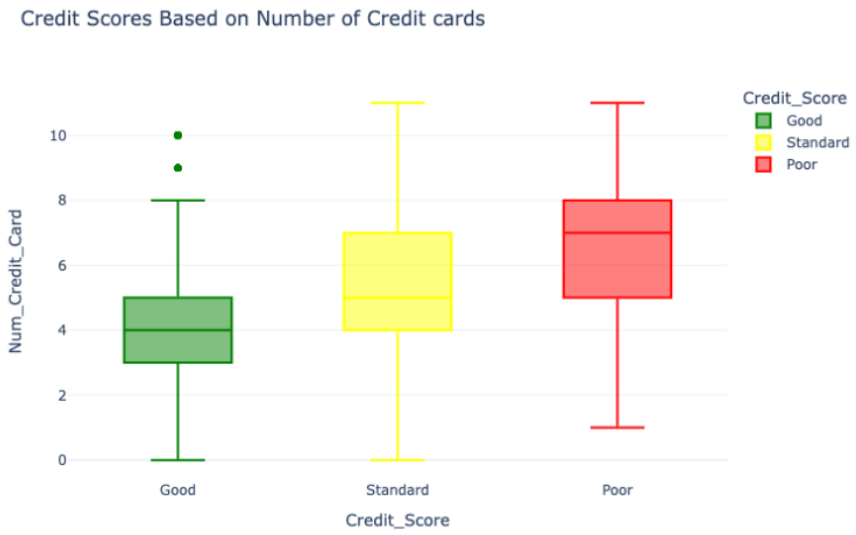
```
fig = px.box(data,
```

```
            x="Credit_Score",
            y="Num_Bank_Accounts",
            color="Credit_Score",
            title="Credit Scores Based on Number o
            color_discrete_map={'Poor':'red',
                                'Standard':'yellow
                                'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



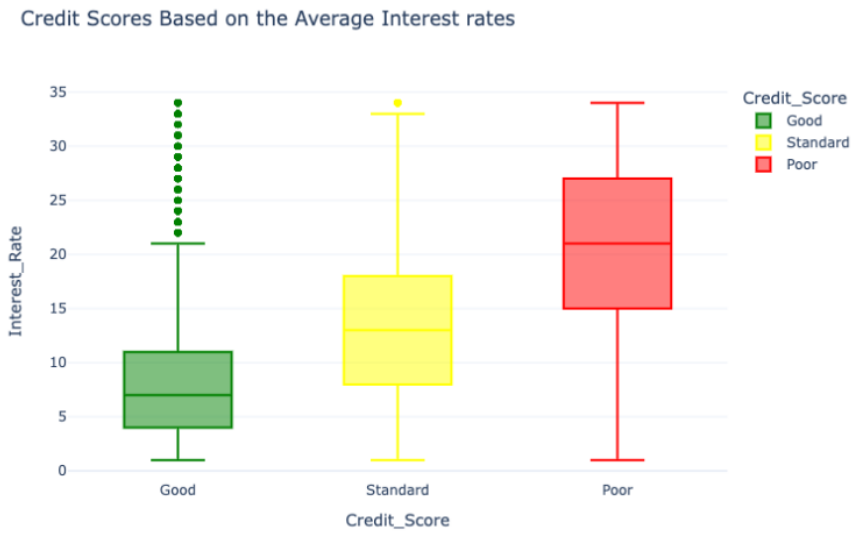
Maintaining more than five accounts is not good for having a good credit score. A person should have 2 – 3 bank accounts only. So having more bank accounts doesn't positively impact credit scores. Now let's see the impact on credit scores based on the number of credit cards you have:

```
fig = px.box(data,
            x="Credit_Score",
            y="Num_Credit_Card",
            color="Credit_Score",
            title="Credit Scores Based on Number o
            color_discrete_map={'Poor':'red',
                                'Standard':'yellow
                                'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



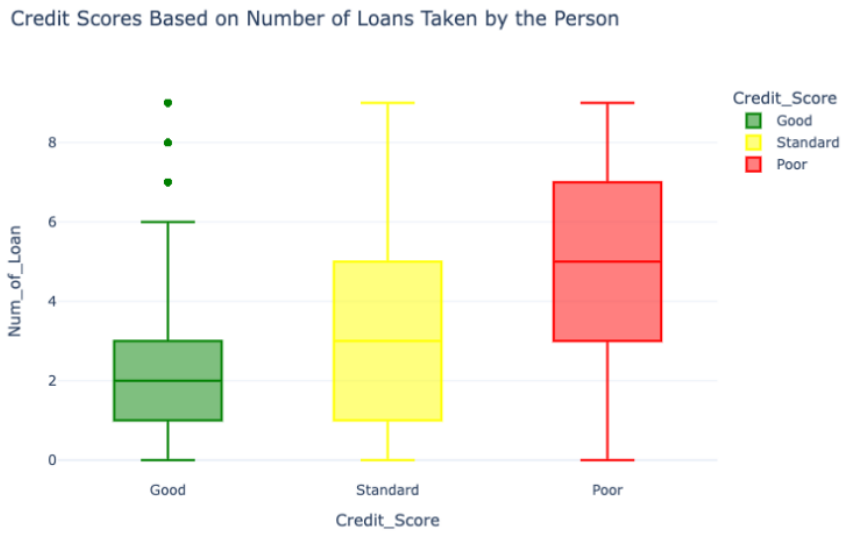
Just like the number of bank accounts, having more credit cards will not positively impact your credit scores. Having 3 – 5 credit cards is good for your credit score. Now let's see the impact on credit scores based on how much average interest you pay on loans and EMIs:

```
fig = px.box(data,
             x="Credit_Score",
             y="Interest_Rate",
             color="Credit_Score",
             title="Credit Scores Based on the Aver
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



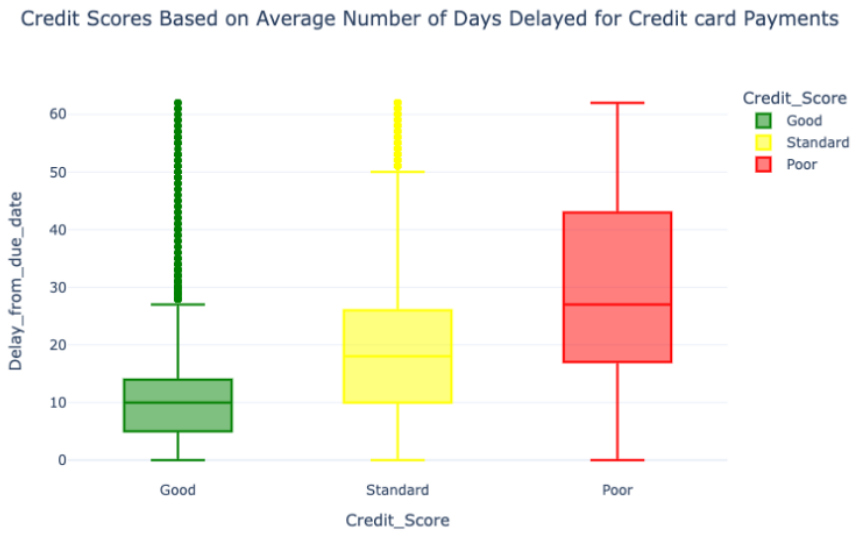
If the average interest rate is 4 – 11%, the credit score is good. Having an average interest rate of more than 15% is bad for your credit scores. Now let's see how many loans you can take at a time for a good credit score:


```
fig = px.box(data,
             x="Credit_Score",
             y="Num_of_Loan",
             color="Credit_Score",
             title="Credit Scores Based on Number o
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



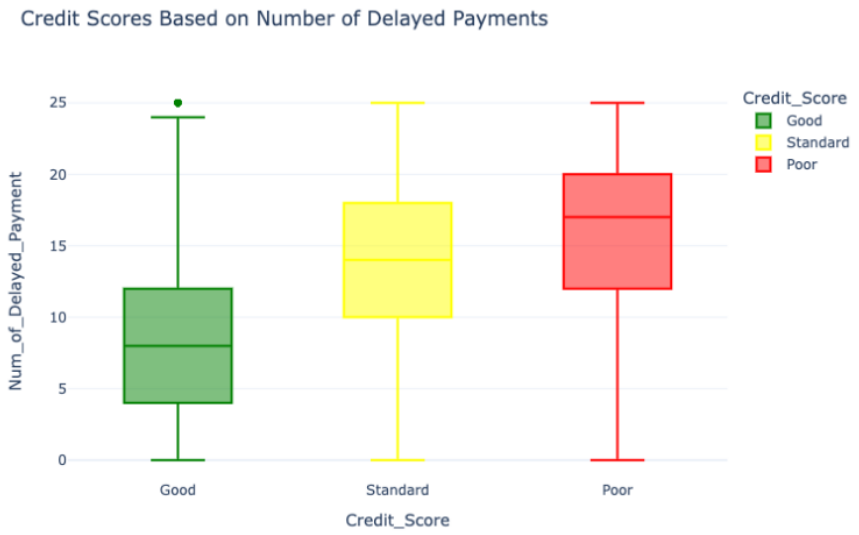
To have a good credit score, you should not take more than 1 – 3 loans at a time. Having more than three loans at a time will negatively impact your credit scores. Now let's see if delaying payments on the due date impacts your credit scores or not:

```
fig = px.box(data,
             x="Credit_Score",
             y="Delay_from_due_date",
             color="Credit_Score",
             title="Credit Scores Based on Average
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



So you can delay your credit card payment 5 – 14 days from the due date. Delaying your payments for more than 17 days from the due date will impact your credit scores negatively. Now let's have a look at if frequently delaying payments will impact credit scores or not:

```
fig = px.box(data,
             x="Credit_Score",
             y="Num_of_Delayed_Payment",
             color="Credit_Score",
             title="Credit Scores Based on Number o
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```

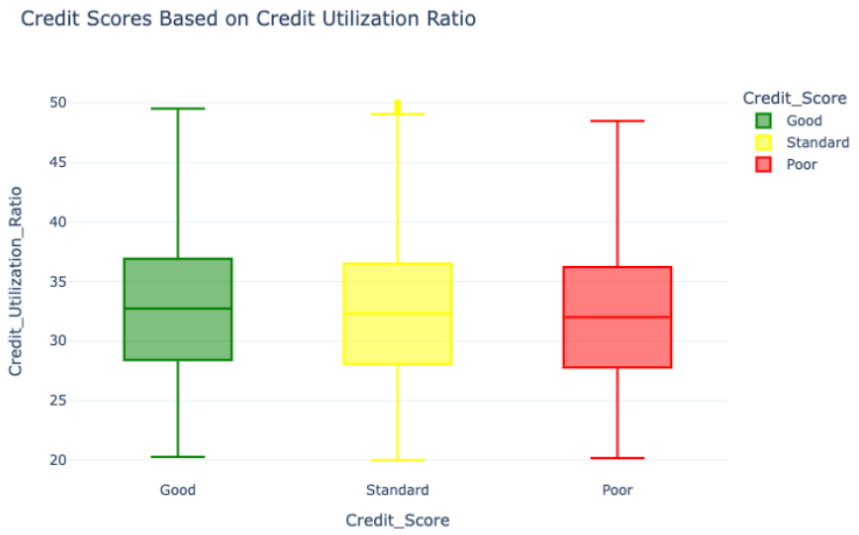


So delaying 4 – 12 payments from the due date will not affect your credit scores. But delaying more than 12 payments from the due date will affect your credit scores negatively. Now let's see if having more debt will affect credit scores or not:

```
fig = px.box(data,
             x="Credit_Score",
             y="Outstanding_Debt",
             color="Credit_Score",
             title="Credit Scores Based on Outstand
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```

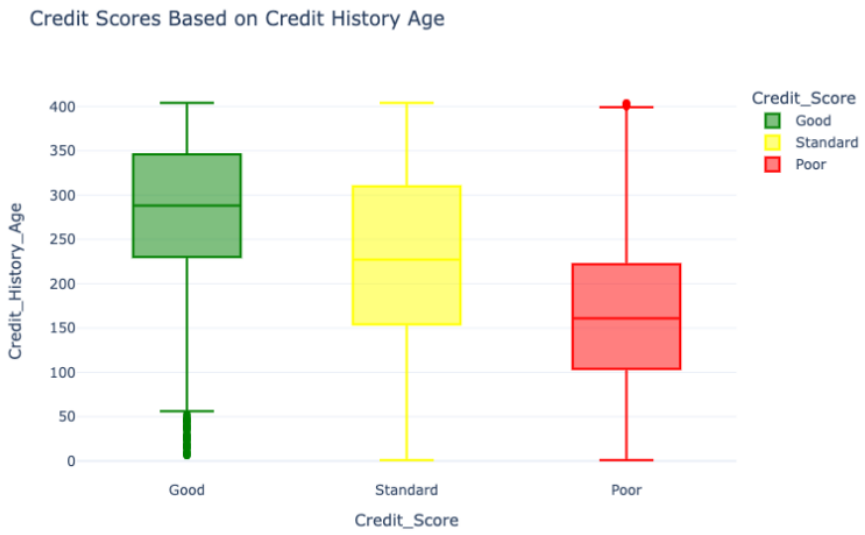
An outstanding debt of $380 – $1150 will not affect your credit scores. But always having a debt of more than $1338 will affect your credit scores negatively. Now let's see if having a high credit utilization ratio will affect credit scores or not:


```
fig = px.box(data,
             x="Credit_Score",
             y="Credit_Utilization_Ratio",
             color="Credit_Score",
             title="Credit Scores Based on Credit U
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



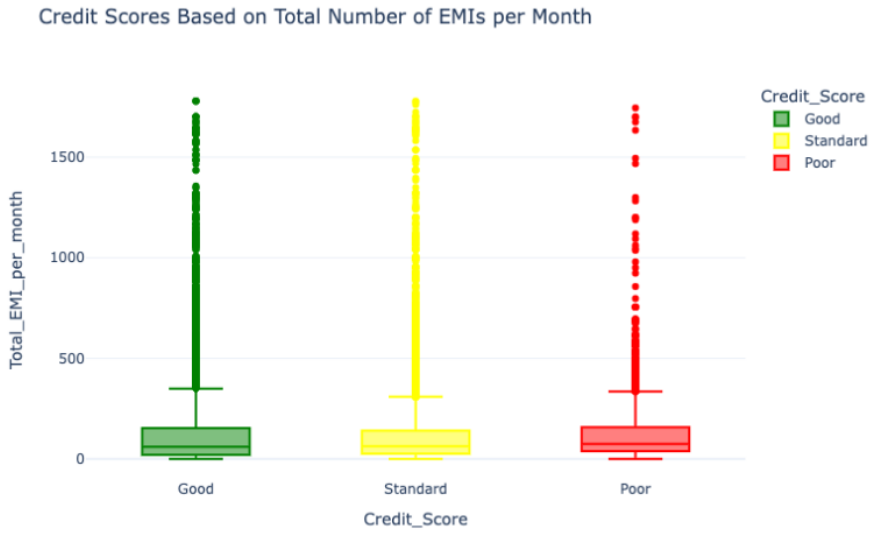
Credit utilization ratio means your total debt divided by your total available credit. According to the above figure, your credit utilization ratio doesn't affect your credit scores. Now let's see how the credit history age of a person affects credit scores:

```
fig = px.box(data,
             x="Credit_Score",
             y="Credit_History_Age",
             color="Credit_Score",
             title="Credit Scores Based on Credit H
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



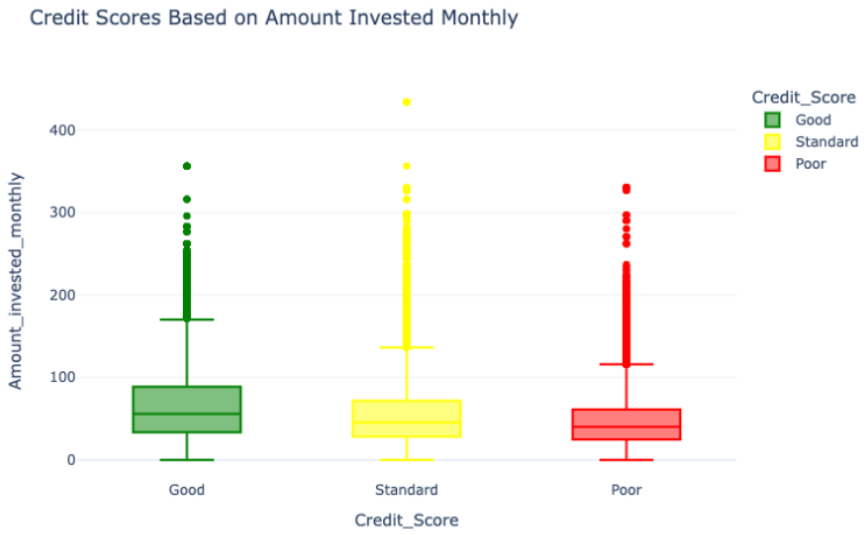
So, having a long credit history results in better credit scores. Now let's see how many EMIs you can have in a month for a good credit score:

```
fig = px.box(data,
             x="Credit_Score",
             y="Total_EMI_per_month",
             color="Credit_Score",
             title="Credit Scores Based on Total Nu
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



The number of EMIs you are paying in a month doesn't affect much on credit scores. Now let's see if your monthly investments affect your credit scores or not:

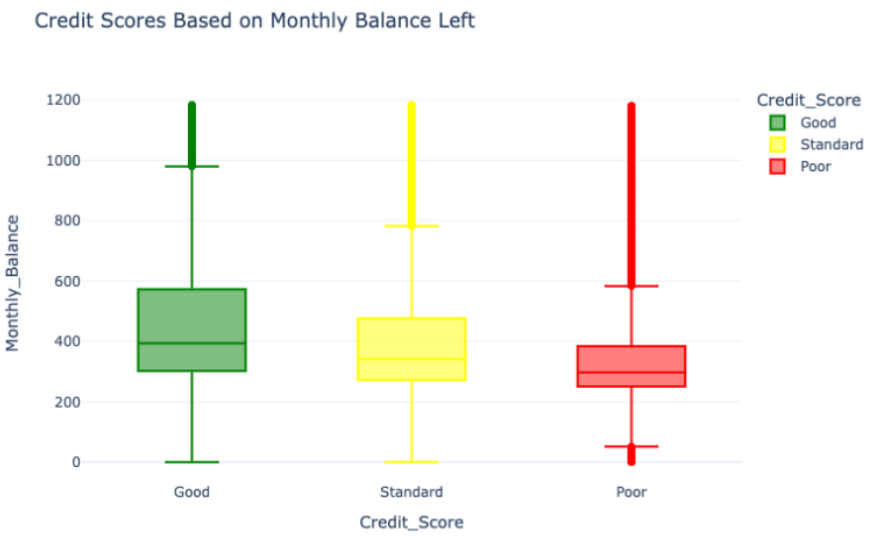
```
fig = px.box(data,
             x="Credit_Score",
             y="Amount_invested_monthly",
             color="Credit_Score",
             title="Credit Scores Based on Amount I
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



The amount of money you invest monthly doesn't affect your credit scores a lot. Now let's see if having a low amount at the end of the month affects credit scores or not:

```
fig = px.box(data,
             x="Credit_Score",
             y="Monthly_Balance",
             color="Credit_Score",
             title="Credit Scores Based on Monthly
             color_discrete_map={'Poor':'red',
                                 'Standard':'yellow
                                 'Good':'green'})
fig.update_traces(quartilemethod="exclusive")
```

```
fig.show()
```



So, having a high monthly balance in your account at the end of the month is good for your credit scores. A monthly balance of less than $250 is bad for credit scores.

Credit Score Classification Model

One more important feature (Credit Mix) in the dataset is valuable for determining credit scores. The credit mix feature tells about the types of credits and loans you have taken.

As the Credit_Mix column is categorical, I will transform it into a numerical feature so that we can use it to train a Machine Learning model for the task of credit score classification:

```
data["Credit_Mix"] = data["Credit_Mix"].map({"Standa
                                "Good": 2,
```

```
                                    "Bad": 0})
```

Now I will split the data into features and labels by selecting the features we found important for our model:

```
from sklearn.model_selection import train_test_spli
x = np.array(data[["Annual_Income", "Monthly_Inhand_
                   "Num_Bank_Accounts", "Num_Credit_
                   "Interest_Rate", "Num_of_Loan",
                   "Delay_from_due_date", "Num_of_De
                   "Credit_Mix", "Outstanding_Debt",
                   "Credit_History_Age", "Monthly_Ba
y = np.array(data[["Credit_Score"]])
```

Now, let's split the data into training and test sets and proceed further by training a credit score classification model:

```
xtrain, xtest, ytrain, ytest = train_test_split(x, y


from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(xtrain, ytrain)
```

Now, let's make predictions from our model by giving inputs to our model according to the features we used to train the model:


```
print("Credit Score Prediction : ")
a = float(input("Annual Income: "))
b = float(input("Monthly Inhand Salary: "))
c = float(input("Number of Bank Accounts: "))
d = float(input("Number of Credit cards: "))
e = float(input("Interest rate: "))
f = float(input("Number of Loans: "))
```

```
g = float(input("Average number of days delayed by
h = float(input("Number of delayed payments: "))
i = input("Credit Mix (Bad: 0, Standard: 1, Good: 3
j = float(input("Outstanding Debt: "))
k = float(input("Credit History Age: "))
l = float(input("Monthly Balance: "))

features = np.array([[a, b, c, d, e, f, g, h, i, j,
print("Predicted Credit Score = ", model.predict(fe
```

```
Credit Score Prediction :
Annual Income: 19114.12
Monthly Inhand Salary: 1824.843333
Number of Bank Accounts: 2
Number of Credit cards: 2
Interest rate: 9
Number of Loans: 2
Average number of days delayed by the person: 12
Number of delayed payments: 3
Credit Mix (Bad: 0, Standard: 1, Good: 3) : 3
Outstanding Debt: 250
Credit History Age: 200
Monthly Balance: 310
Predicted Credit Score =  ['Good']
```

So this is how you can use Machine Learning for the task of Credit Score Classification using Python.

Summary

Classifying customers based on their credit scores helps banks and credit card companies immediately to issue loans to customers with good creditworthiness. A person with a good credit score will get loans from any bank and financial institution. I hope you liked this article on Credit Score Classification with Machine Learning using Python. Feel free to ask valuable questions in the comments section below.

Aman Kharwal

I'm a writer and data scientist on a mission to educate others about the incredible power of data📈.

ARTICLES: 1354

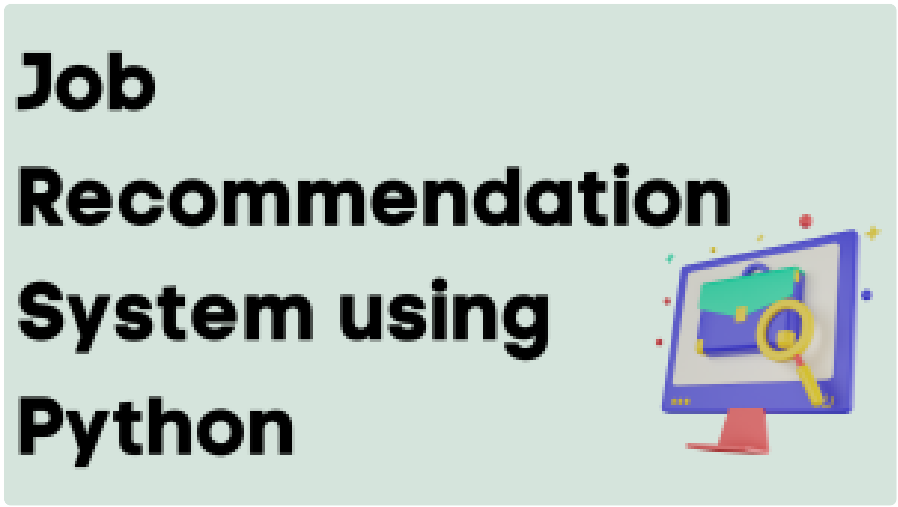

PREVIOUS

NEXT


Recommended For You

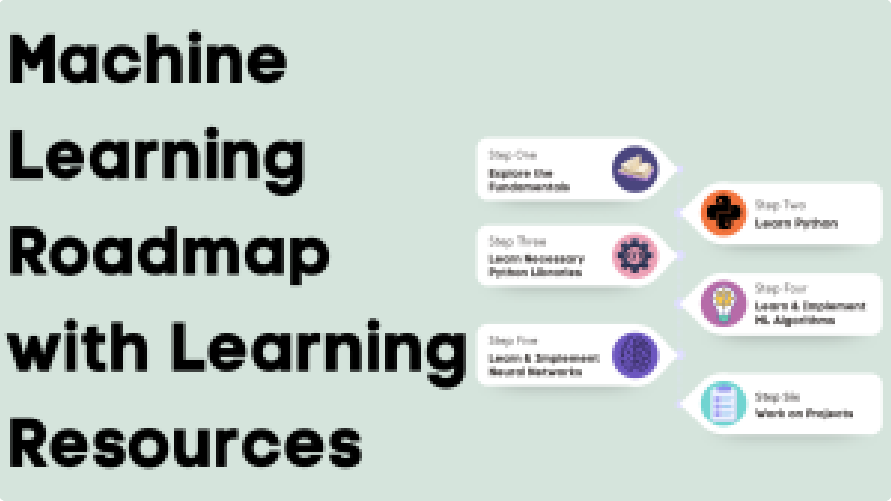
Food Delivery Time Prediction using Python

January 2, 2023



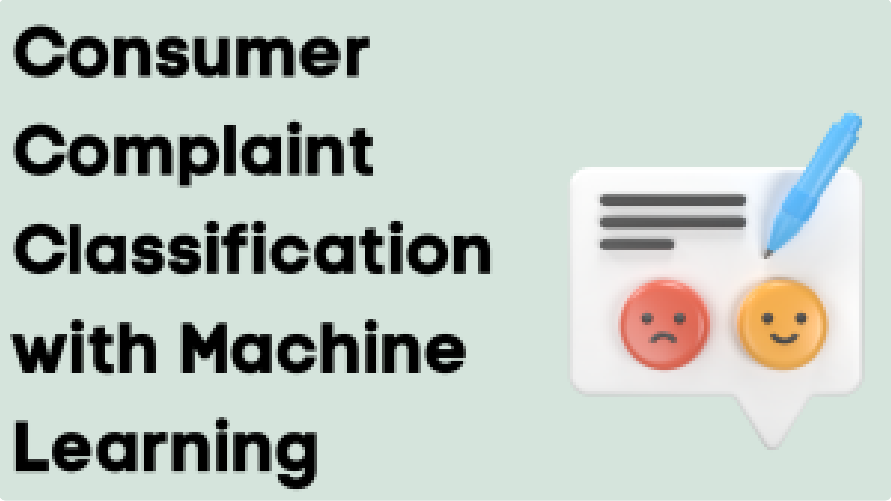
Job Recommendation System using Python

December 12, 2022



Machine Learning Roadmap

December 6, 2022 / 2 Comments



Consumer Complaint Classification with Machine Learning

November 28, 2022

Leave a Reply

Enter your comment here...

FACEBOOK INSTAGRAM MEDIUM LINKEDIN