CREDIT SCORE CLASSIFICATION

April 30, 2023

1 Problem Satetment :-) Credit Score Classification: Case Study

Description :-) The credit score of a person determines the creditworthiness of the person. It helps financial companies determine if you can repay the loan or credit you are applying for.

1.0.1 1. Importing Necessary Libraries

[1]:
```
import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
import matplotlib.pyplot as plt
```

2 2. Data Set Information

1. ID: Unique ID of the record
2. Customer_ID: Unique ID of the customer
3. Month: Month of the year
4. Name: The name of the person
5. Age: The age of the person
6. SSN: Social Security Number of the person
7. Occupation:The occupation of the person
8. Annual_Income: The Annual Income of the person
9. Monthly_Inhand_Salary: Monthly in-hand salary of the person
10. Num_Bank_Accounts: The number of bank accounts of the person
11. Num_Credit_Card: Number of credit cards the person is having
12. Interest_Rate: The interest rate on the credit card of the person
13. Num_of_Loan: The number of loans taken by the person from the bank
14. Type_of_Loan: The types of loans taken by the person from the bank
15. Delay_from_due_date: The average number of days delayed by the person from the date of payment
16. Num_of_Delayed_Payment: Number of payments delayed by the person
17. Changed_Credit_Card: The percentage change in the credit card limit of the person
18. Num_Credit_Inquiries: The number of credit card inquiries by the person
19. Credit_Mix: Classification of Credit Mix of the customer

20. Outstanding_Debt: The outstanding balance of the person
21. Credit_Utilization_Ratio: The credit utilization ratio of the credit card of the customer
22. Credit_History_Age: The age of the credit history of the person
23. Payment_of_Min_Amount: Yes if the person paid the minimum amount to be paid only, otherwise no.
24. Total_EMI_per_month: The total EMI per month of the person
25. Amount_invested_monthly: The monthly amount invested by the person
26. Payment_Behaviour:The payment behaviour of the person
27. Monthly_Balance: The monthly balance left in the account of the person
28. Credit_Score: The credit score of the person

- The Credit_Score column is the target variable in this problem. You are required to find relationships based on how banks classify credit scores and train a model to classify the credit score of a person.

2.0.1 2.1 Reading Datset

[2]:
```
data=pd.read_csv("train.csv")
data.head()
```

[2]:

	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:

[3]:
```
data.shape
```

[3]: (100000, 28)

- The DataSet Contain Observation => 100000 , Feature => 28

[4]:
```
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
```

- Some Feature is Object and some numeric

[5]:
```
data.describe()
```

[5]:

	ID	Customer_ID	Month	Age	\
count	100000.000000	100000.000000	100000.000000	100000.000000	
mean	80631.500000	25982.666640	4.500000	33.316340	
std	43301.486619	14340.543051	2.291299	10.764812	
min	5634.000000	1006.000000	1.000000	14.000000	
25%	43132.750000	13664.500000	2.750000	24.000000	
50%	80631.500000	25777.000000	4.500000	33.000000	
75%	118130.250000	38385.000000	6.250000	42.000000	
max	155629.000000	50999.000000	8.000000	56.000000	

	SSN	Annual_Income	Monthly_Inhand_Salary	Num_Bank_Accounts \
count	1.000000e+05	100000.000000	100000.000000	100000.000000
mean	5.004617e+08	50505.123449	4197.270835	5.368820
std	2.908267e+08	38299.422093	3186.432497	2.593314
min	8.134900e+04	7005.930000	303.645417	0.000000
25%	2.451686e+08	19342.972500	1626.594167	3.000000
50%	5.006886e+08	36999.705000	3095.905000	5.000000
75%	7.560027e+08	71683.470000	5957.715000	7.000000
max	9.999934e+08	179987.280000	15204.633333	11.000000

	Num_Credit_Card	Interest_Rate	...	Delay_from_due_date	\
count	100000.000000	100000.00000	...	100000.00000	
mean	5.533570	14.53208	...	21.08141	
std	2.067098	8.74133	...	14.80456	
min	0.000000	1.00000	...	0.00000	
25%	4.000000	7.00000	...	10.00000	
50%	5.000000	13.00000	...	18.00000	
75%	7.000000	20.00000	...	28.00000	
max	11.000000	34.00000	...	62.00000	

	Num_of_Delayed_Payment	Changed_Credit_Limit	Num_Credit_Inquiries	\
count	100000.000000	100000.000000	100000.000000	
mean	13.313120	10.470323	5.798250	
std	6.237166	6.609481	3.867826	
min	0.000000	0.500000	0.000000	
25%	9.000000	5.380000	3.000000	
50%	14.000000	9.400000	5.000000	

75%	18.000000	14.850000	8.000000
max	25.000000	29.980000	17.000000

	Outstanding_Debt	Credit_Utilization_Ratio	Credit_History_Age	\
count	100000.000000	100000.000000	100000.000000	
mean	1426.220376	32.285173	221.220460	
std	1155.129026	5.116875	99.680716	
min	0.230000	20.000000	1.000000	
25%	566.072500	28.052567	144.000000	
50%	1166.155000	32.305784	219.000000	
75%	1945.962500	36.496663	302.000000	
max	4998.070000	50.000000	404.000000	

	Total_EMI_per_month	Amount_invested_monthly	Monthly_Balance
count	100000.000000	100000.000000	100000.000000
mean	107.699208	55.101315	392.697586
std	132.267056	39.006932	201.652719
min	0.000000	0.000000	0.007760
25%	29.268886	27.959111	267.615983
50%	66.462304	45.156550	333.865366
75%	147.392573	71.295797	463.215683
max	1779.103254	434.191089	1183.930696

[8 rows x 21 columns]

3 3. Handling Null Value

[6]:
```
data.isnull()
```

[6]:

	ID	Customer_ID	Month	Name	Age	SSN	Occupation	\
0	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	
99995	False	False	False	False	False	False	False	
99996	False	False	False	False	False	False	False	
99997	False	False	False	False	False	False	False	
99998	False	False	False	False	False	False	False	
99999	False	False	False	False	False	False	False	

	Annual_Income	Monthly_Inhand_Salary	Num_Bank_Accounts	... \
0	False	False	False	...
1	False	False	False	...
2	False	False	False	...

```
3              False                  False                  False    ...
4              False                  False                  False    ...
...              ...                    ...                    ... ...
99995          False                  False                  False    ...
99996          False                  False                  False    ...
99997          False                  False                  False    ...
99998          False                  False                  False    ...
99999          False                  False                  False    ...

       Credit_Mix  Outstanding_Debt  Credit_Utilization_Ratio      \
0           False             False                     False
1           False             False                     False
2           False             False                     False
3           False             False                     False
4           False             False                     False
...           ...               ...                       ...
99995       False             False                     False
99996       False             False                     False
99997       False             False                     False
99998       False             False                     False
99999       False             False                     False

       Credit_History_Age  Payment_of_Min_Amount  Total_EMI_per_month  \
0                   False                  False                False
1                   False                  False                False
2                   False                  False                False
3                   False                  False                False
4                   False                  False                False
...                   ...                    ...                  ...
99995               False                  False                False
99996               False                  False                False
99997               False                  False                False
99998               False                  False                False
99999               False                  False                False

       Amount_invested_monthly  Payment_Behaviour  Monthly_Balance  \
0                        False              False            False
1                        False              False            False
2                        False              False            False
3                        False              False            False
4                        False              False            False
...                        ...                ...              ...
99995                    False              False            False
99996                    False              False            False
99997                    False              False            False
99998                    False              False            False
99999                    False              False            False
```

	Credit_Score
0	False
1	False
2	False
3	False
4	False
...	...
99995	False
99996	False
99997	False
99998	False
99999	False

[100000 rows x 28 columns]

[7]:
```
data.isnull().sum()
```

[7]:
```
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:

[8]:
```
data['Credit_Score'].value_counts()
```

[8]:
```
Standard    53174
Poor        28998
Good        17828
Name: Credit_Score, dtype: int64
```

4 4. Data Exploration

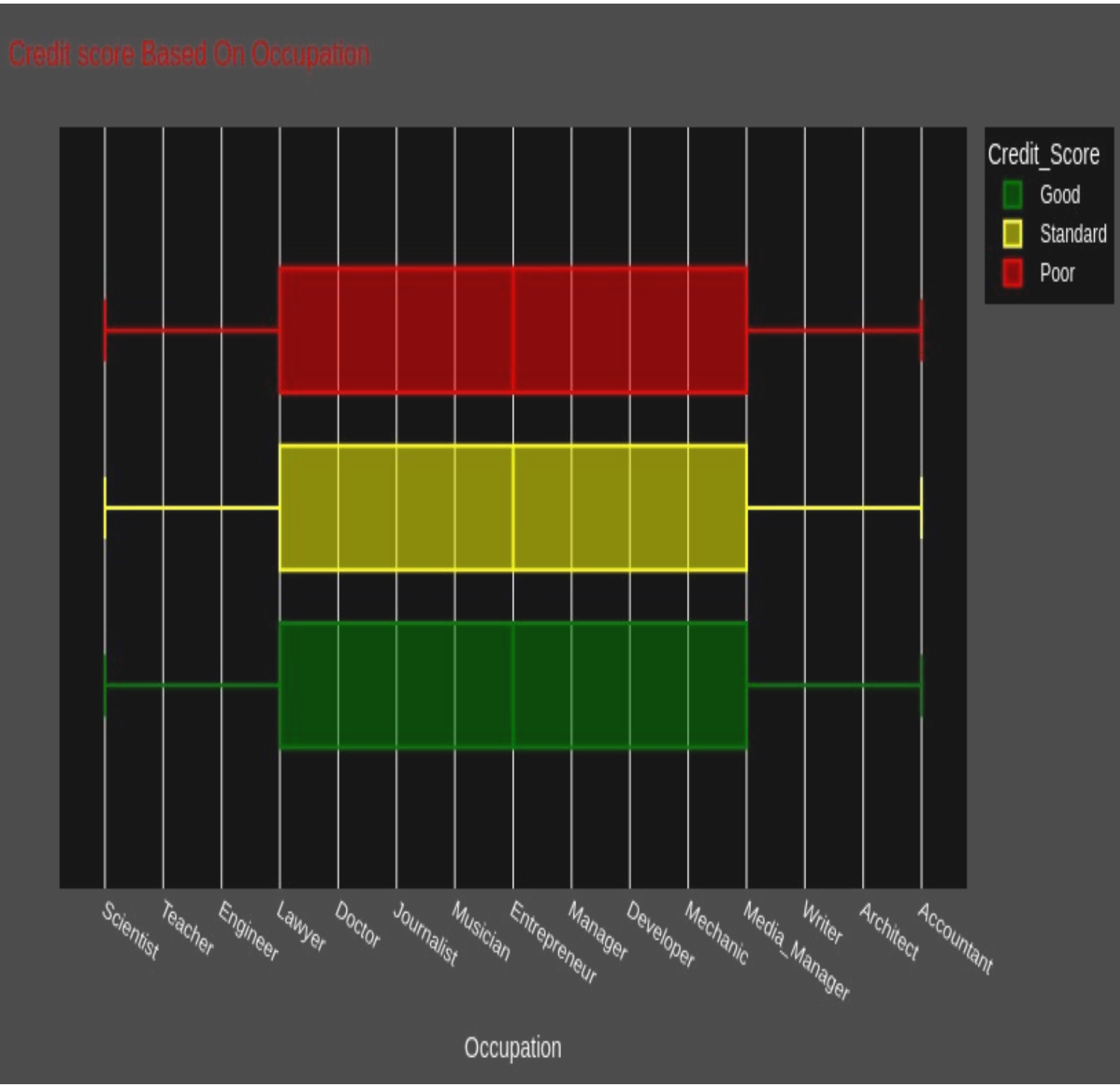
- The dataset has many features that can train a Machine Learning model for credit score classification.Let's explore all the features one by one.

[9]:
```
data.columns.unique()
```

[9]:
```
Index(['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',
       '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'],
      dtype='object')
```

[10]:
```
fig=px.box(data,x='Occupation',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'}
                )
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
   gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
   dark gray
    font_color='white',
    title={'text':'Credit score Based On Occupation','font_color':'red'}
)
fig.show()
```

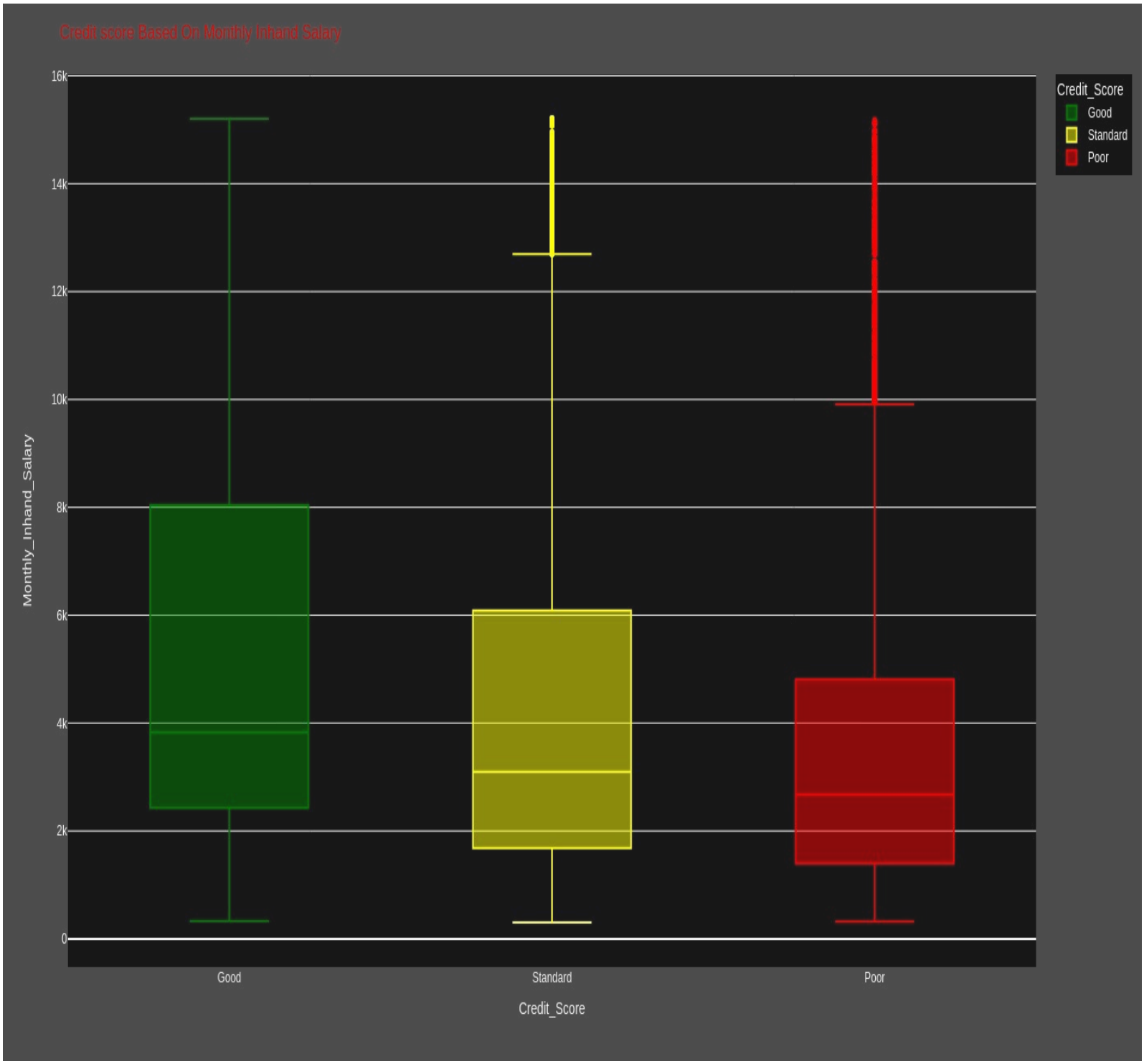
- There's not much difference in the credit scores of all occupations mentioned in the data. Now let's explore whether the Annual Income of the person impacts your credit scores or not:


Credit score Based On Occupation
Credit_Score
Good
Standard
Poor
Scientist
Teacher
Engineer
Lawyer
Doctor
Journalist
Musician
Entrepreneur
Manager
Developer
Mechanic
Media_Manager
Writer
Architect
Accountant
Occupation

[11]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Annual_Income',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Occupation','font_color':'red'},
    width=1400,
    height=900
)
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

[12]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Monthly_Inhand_Salary',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Monthly Inhand Salary ','font_color':
  'red'},
    width=1400,
    height=900
)
fig.show()
```

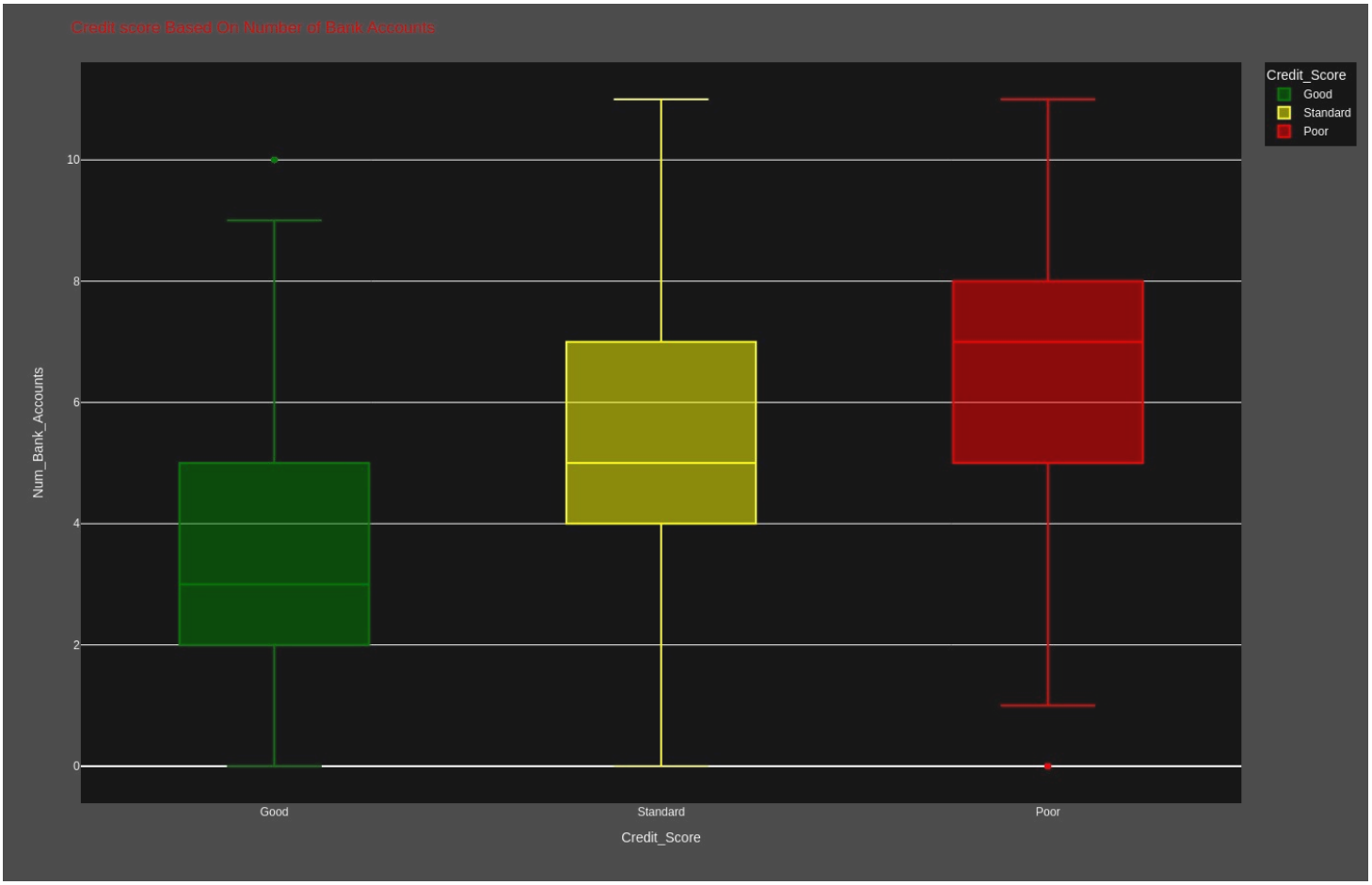

Credit score Based On Monthly Inhand Salary
16k
14k
12k
10k
8k
6k
4k
2k
0
Monthly_Inhand_Salary
Good
Standard
Poor
Credit_Score
Credit_Score
Good
Standard
Poor

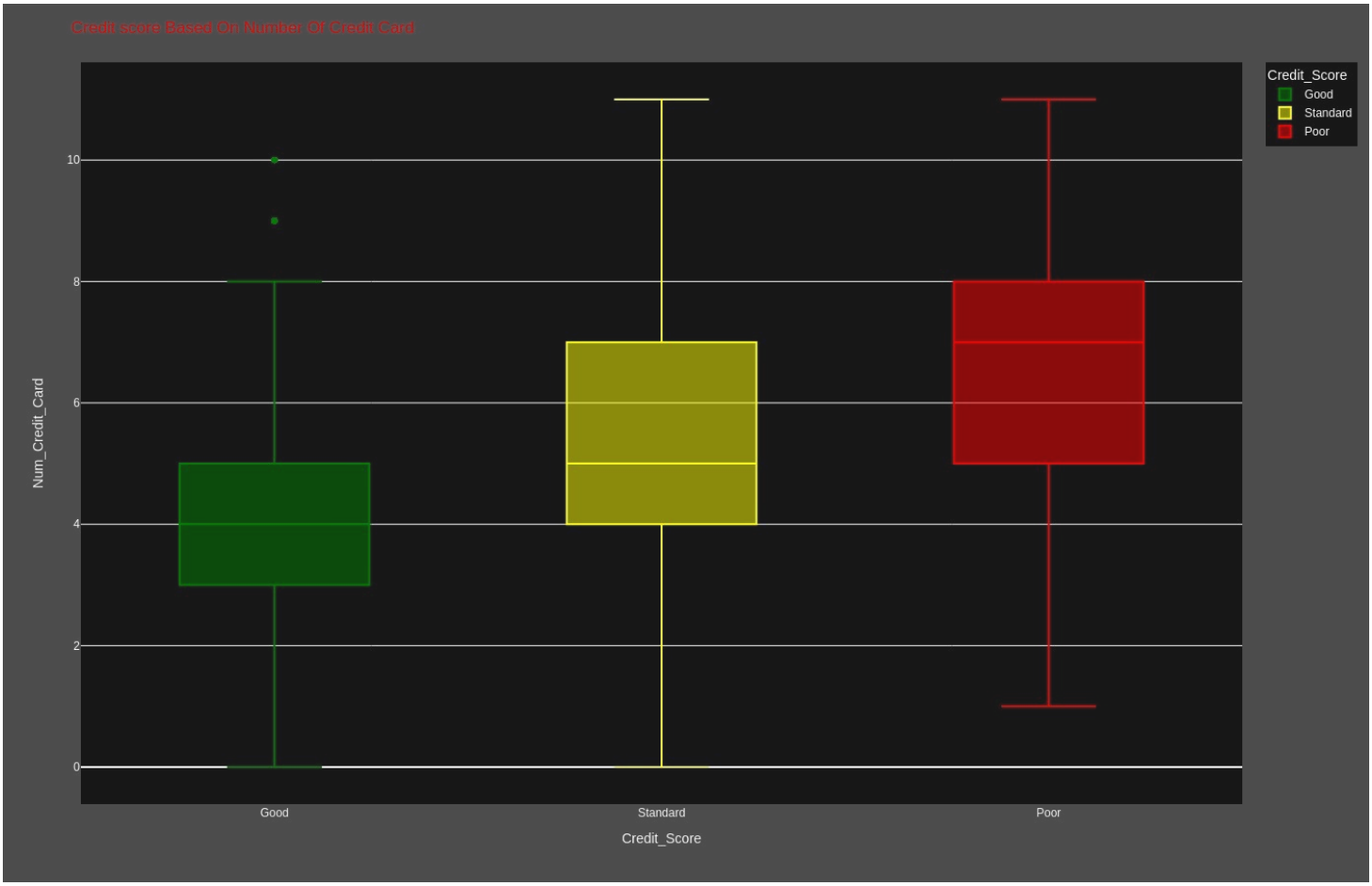
- Like annual income, the more monthly in-hand salary you earn, the better your credit score will become.
- Now let's see if having more bank accounts impacts credit scores or not

[13]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Num_Bank_Accounts',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Number of Bank Accounts ','font_color':
  'red'},
    width=1400,
    height=900
)
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:

[14]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Num_Credit_Card',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
```


Credit score Based On Number of Bank Accounts
Credit_Score
Good
Standard
Poor
Num_Bank_Accounts
0
2
4
6
8
10
Good
Standard
Poor
Credit_Score


Credit score Based On Number Of Credit Card
Credit_Score
Good
Standard
Poor
Num_Credit_Card
0
2
4
6
8
10
Good
Standard
Poor
Credit_Score

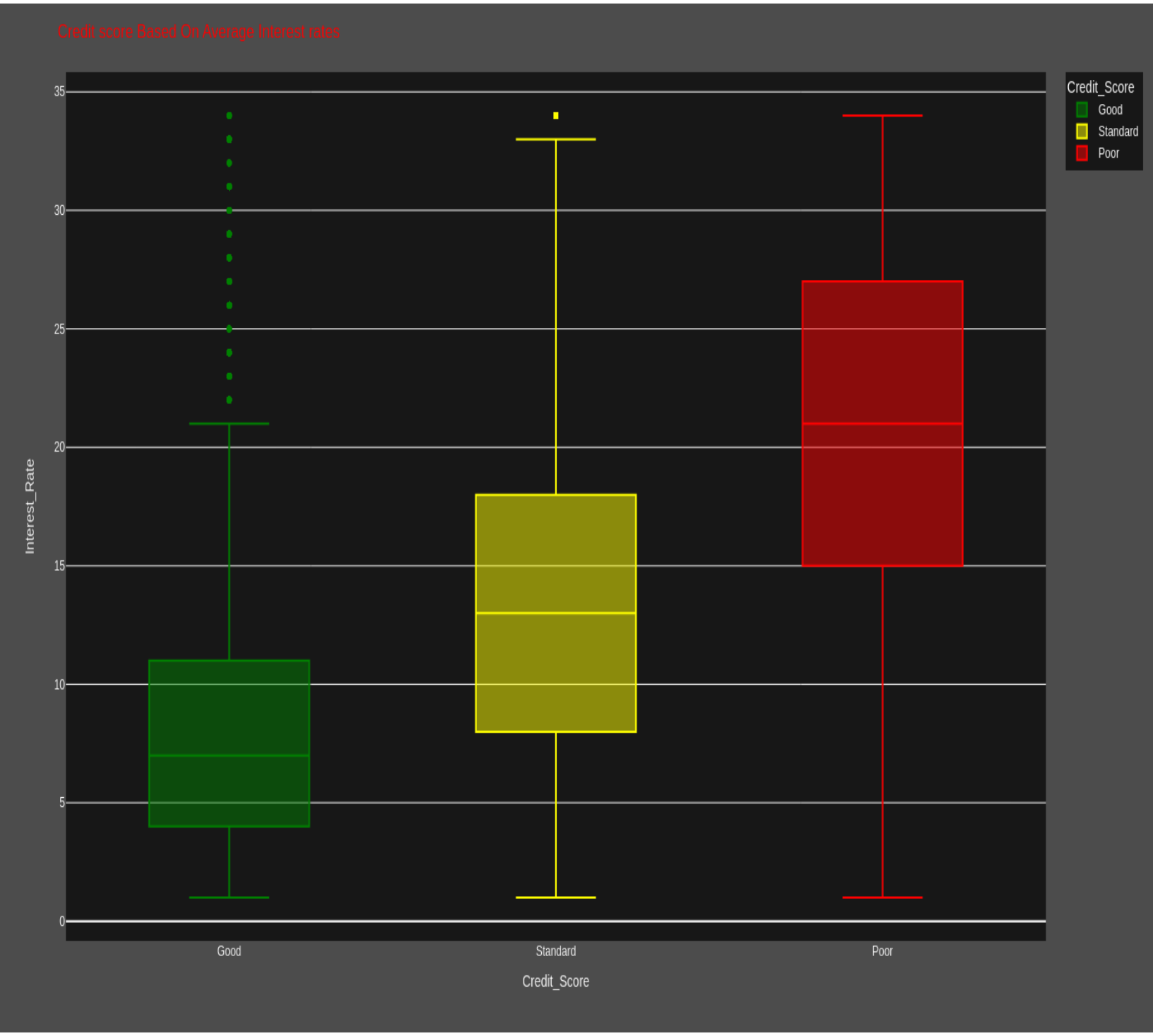
```
    title={'text':'Credit score Based On Number Of Credit Card','font_color':
  'red'},
    width=1400,
    height=900
)
fig.show()
```

- 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:

[15]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Interest_Rate',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Average Interest rates','font_color':
  'red'},
    width=1400,
    height=900
)
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:

[16]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Num_of_Loan',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
```


Credit score Based On Average Interest rates
Credit_Score
Good
Standard
Poor
35
30
25
20
15
10
5
0
Interest_Rate
Good
Standard
Poor
Credit_Score

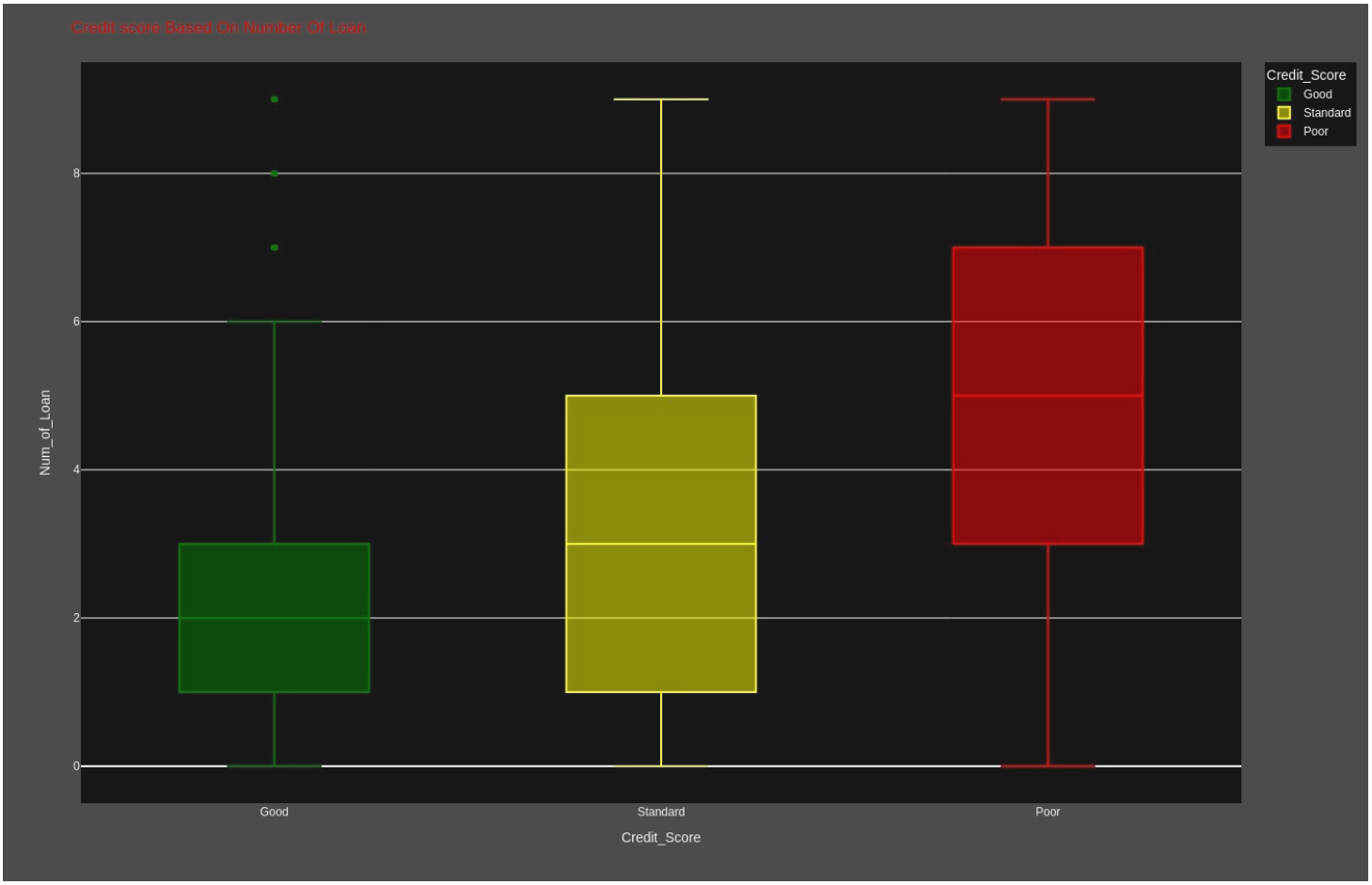
```
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
      plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
   gray
      paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
   dark gray
      font_color='white',
      title={'text':'Credit score Based On Number Of Loan','font_color':'red'},
      width=1400,
      height=900
)
fig.show()
```

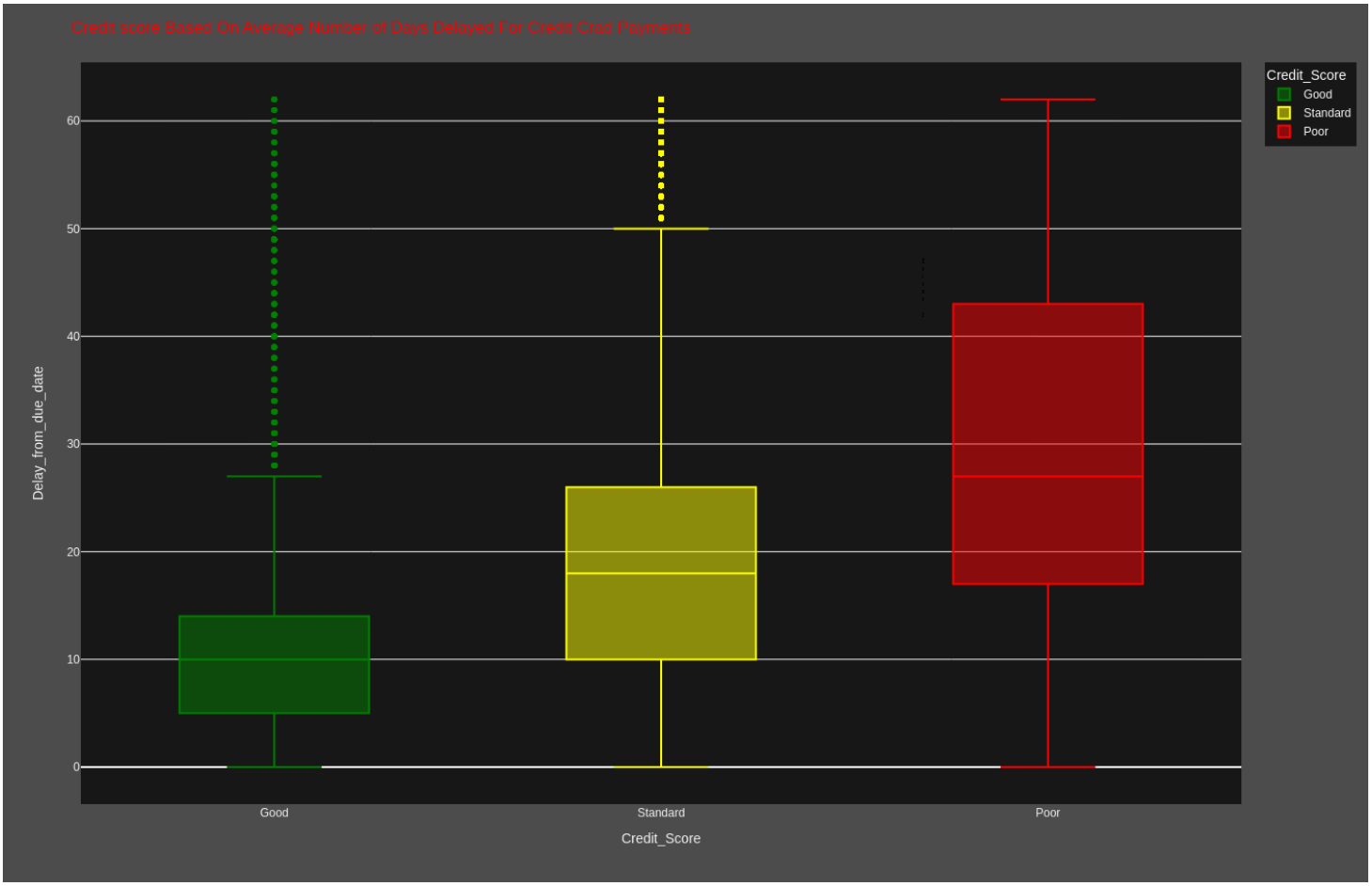
- To have a good credit score, you should not take more than 1 – 3 loans at a time.
- Having more than three loans at a time will negatively impact your credit scores.
- Now let's see if delaying payments on the due date impacts your credit scores or not:

[17]:
```
fig=px.box(data,
                  x='Credit_Score',
                  y='Delay_from_due_date',
                  color='Credit_Score',
                  color_discrete_map={'Poor':'red',
                                      'Standard':'yellow',
                                      'Good':'green'},
                  )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
      plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
   gray
      paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
   dark gray
      font_color='white',
      title={'text':'Credit score Based On Average Number of Days Delayed For␣
   Credit Crad Payments ','font_color':'red'},
      width=1400,
      height=900
)
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:

[18]:
```
fig=px.box(data,
                  x='Credit_Score',
                  y='Num_of_Delayed_Payment',
```


Credit score Based On Number Of Loan
Credit_Score
Good
Standard
Poor
Num_of_Loan
0
2
4
6
8
Good
Standard
Poor
Credit_Score

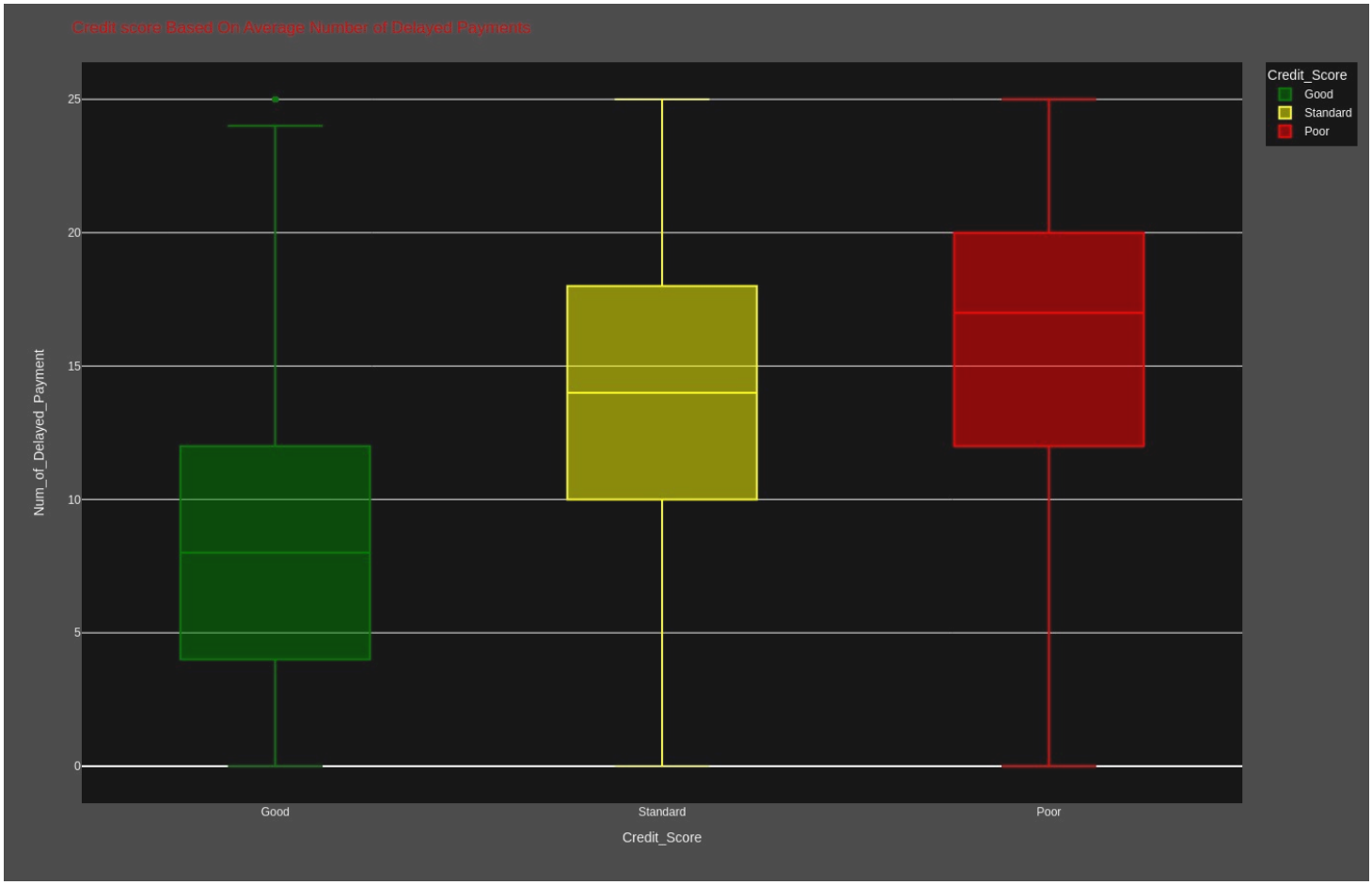

Credit score Based On Average Number of Days Delayed For Credit Cradi Payments
Credit_Score
Good
Standard
Poor
Delay_from_due_date
0
10
20
30
40
50
60
Good
Standard
Poor
Credit_Score

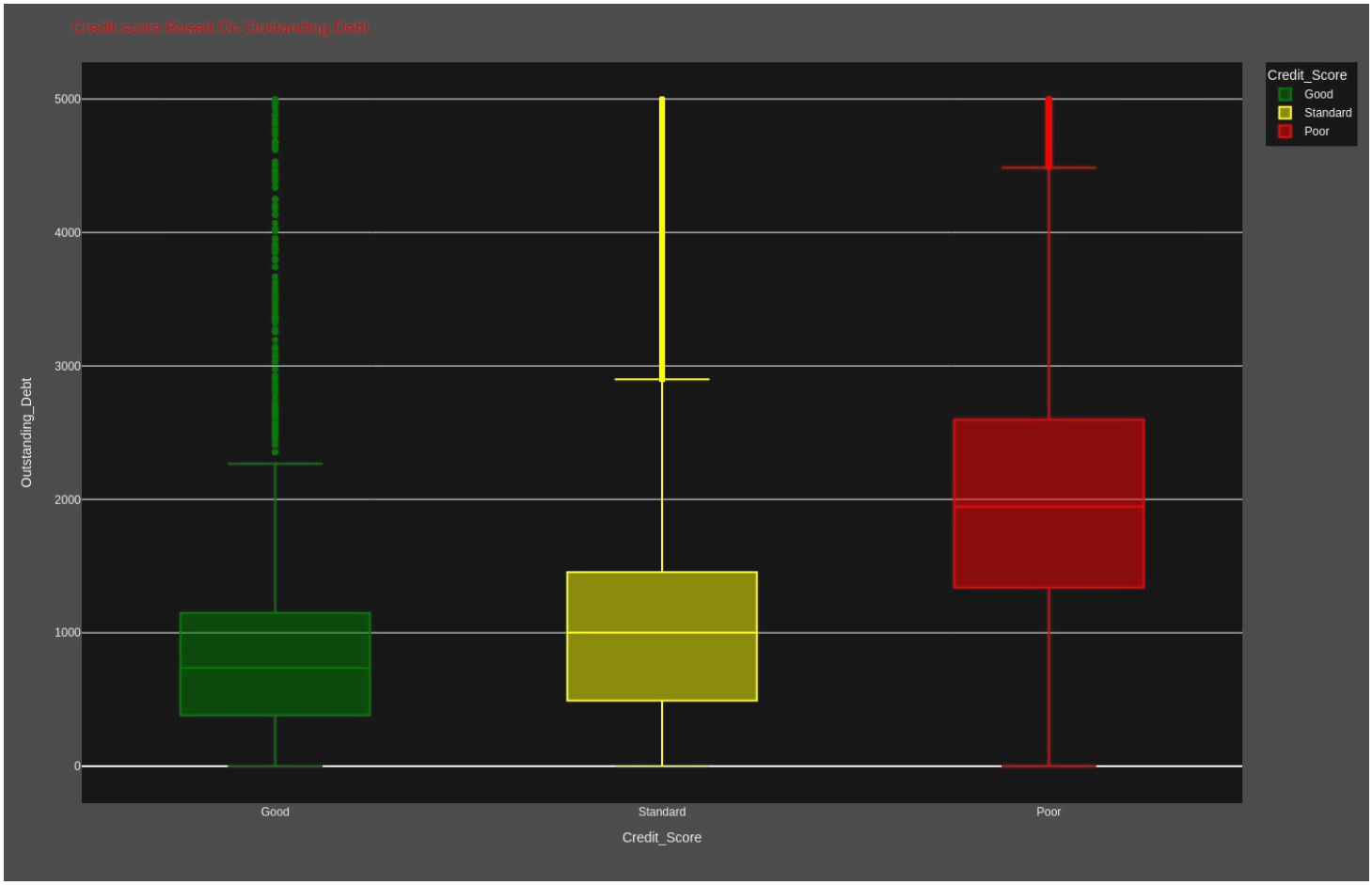
```
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Average Number of Delayed␣
  Payments','font_color':'red'},
    width=1400,
    height=900
)
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:

[19]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Outstanding_Debt',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Oustanding Debt','font_color':'red'},
    width=1400,
    height=900
)
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 nega-


Credit score Based On Average Number of Delayed Payments
Num_of_Delayed_Payment
25
20
15
10
5
0
Good
Standard
Poor
Credit_Score
Credit_Score
Good
Standard
Poor


Credit score Based On Oustanding Debt
Outstanding_Debt
5000
4000
3000
2000
1000
0
Good
Standard
Poor
Credit_Score
Credit_Score
Good
Standard
Poor

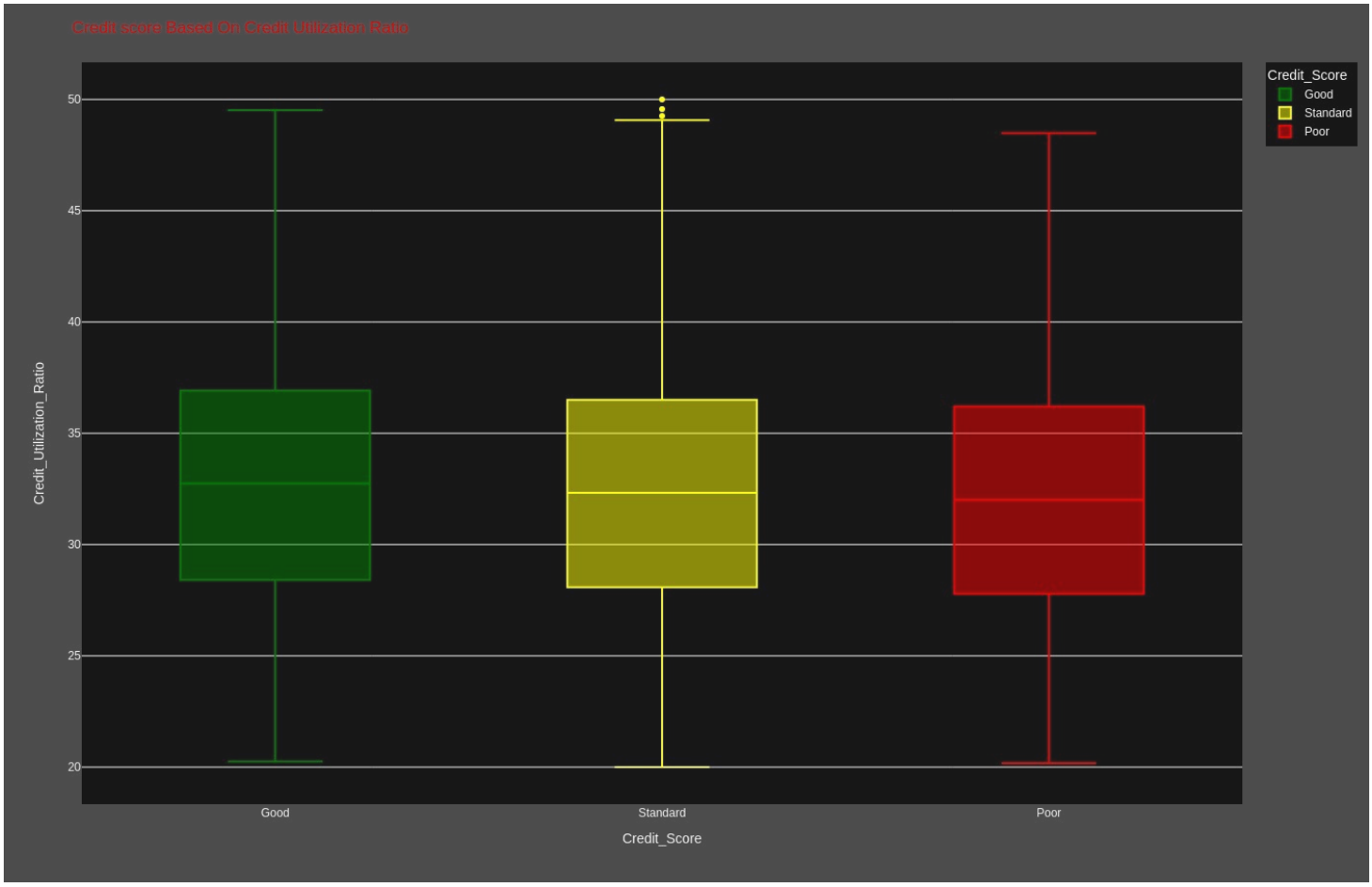
tively.

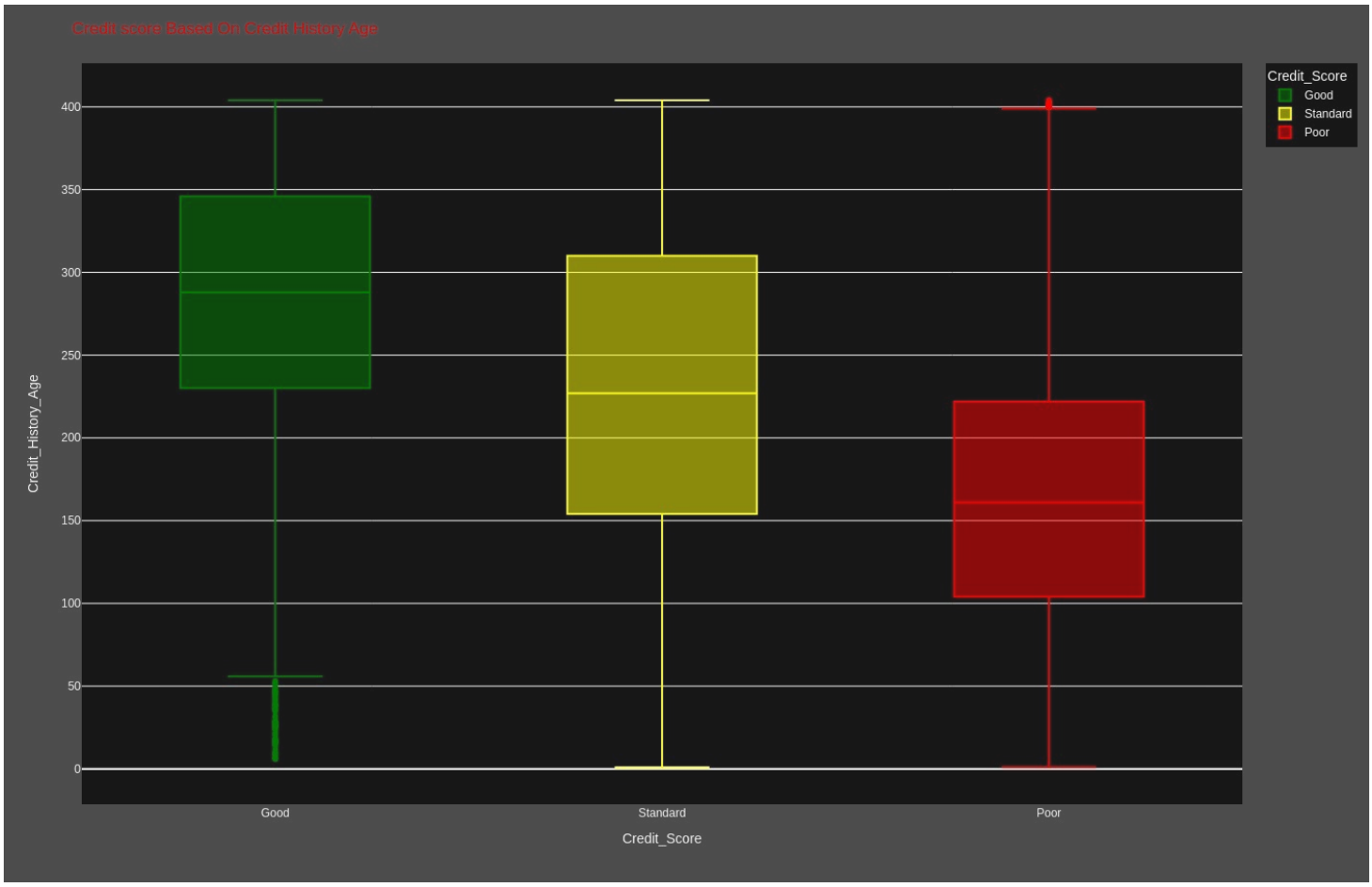
- Now let's see if having a high credit utilization ratio will affect credit scores or not:

[20]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Credit_Utilization_Ratio',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Credit Utilization Ratio␣
  ','font_color':'red'},
    width=1400,
    height=900
)
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:

[21]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Credit_History_Age',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
```


Credit score Based On Credit Utilization Ratio
Credit_Score
Good
Standard
Poor
Credit_Utilization_Ratio
50
45
40
35
30
25
20
Good
Standard
Poor
Credit_Score


Credit score Based On Credit History Age
Credit_Score
Good
Standard
Poor
Credit_History_Age
400
350
300
250
200
150
100
50
0
Good
Standard
Poor
Credit_Score

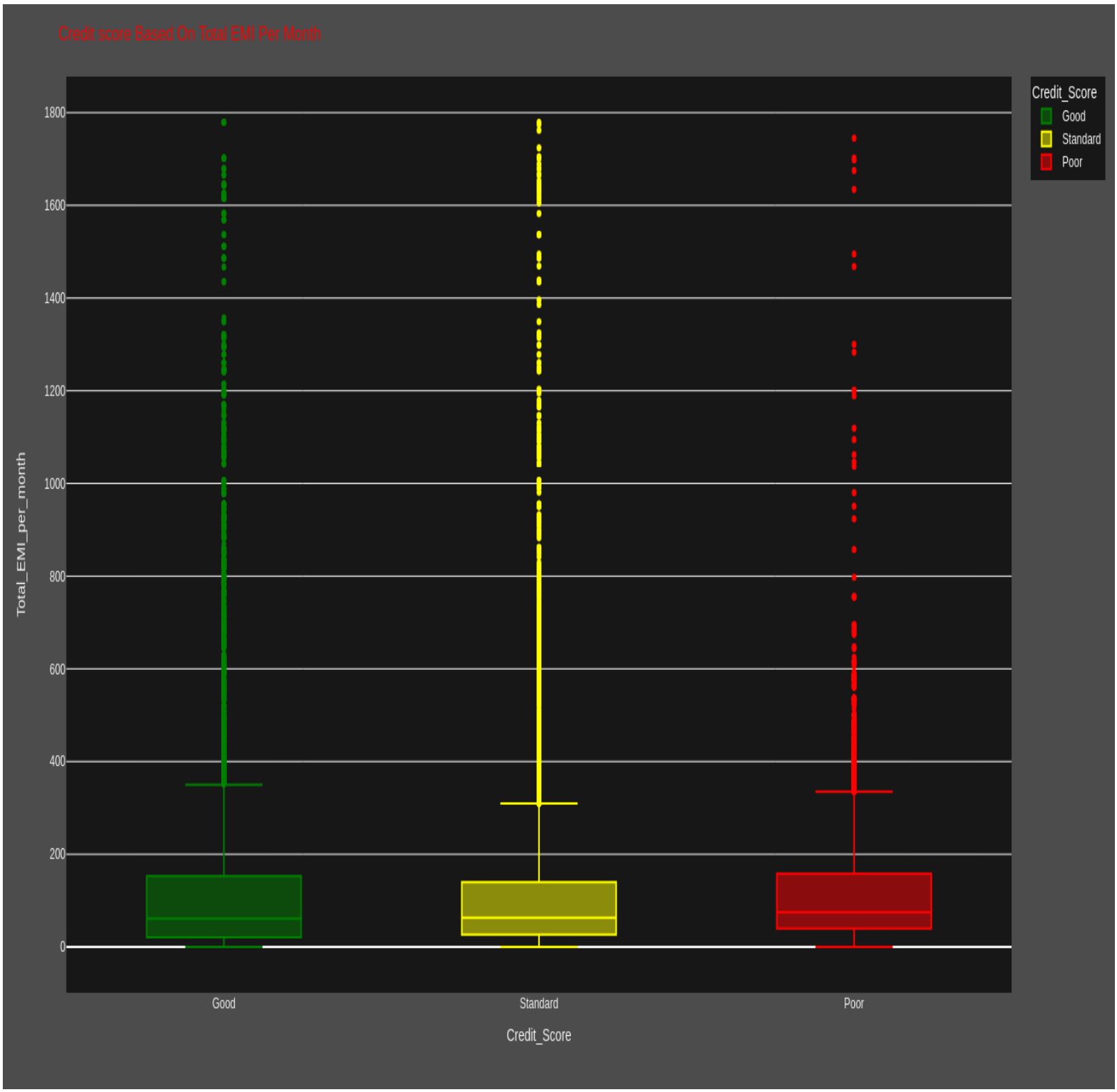
```
    title={'text':'Credit score Based On Credit History Age','font_color':
  'red'},
    width=1400,
    height=900
)
fig.show()
```

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:

[22]:
```
fig=px.box(data,
           x='Credit_Score',
           y='Total_EMI_per_month',
           color='Credit_Score',
           color_discrete_map={'Poor':'red',
                               'Standard':'yellow',
                               'Good':'green'},
           )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Total EMI Per Month','font_color':
  'red'},
    width=1400,
    height=900
)
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:

[23]:
```
fig=px.box(data,
           x='Credit_Score',
           y='Amount_invested_monthly',
           color='Credit_Score',
           color_discrete_map={'Poor':'red',
                               'Standard':'yellow',
                               'Good':'green'},
           )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
```


Credit score Based On Total EMI Per Month
Credit_Score
Good
Standard
Poor
1800
1600
1400
1200
1000
800
600
400
200
0
Total_EMI_per_month
Good
Standard
Poor
Credit_Score

```
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Amount Invested Monthly','font_color':
  'red'},
    width=1400,
    height=900
)
fig.show()
```

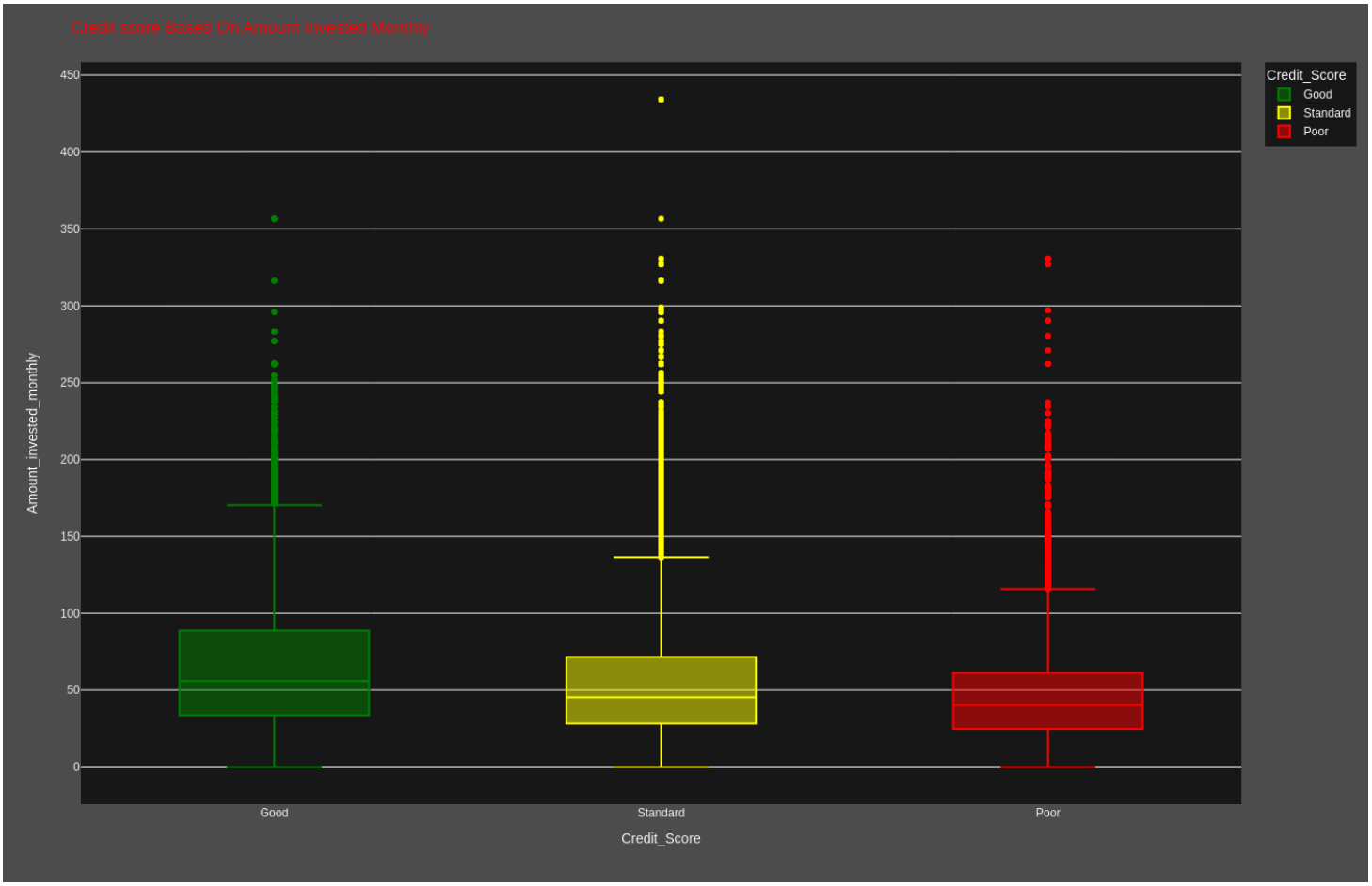
- The amount of money you invest monthly doesn't affect your credit scores a lot.
- Now let's see if having a low amount at the end of the month affects credit scores or not:

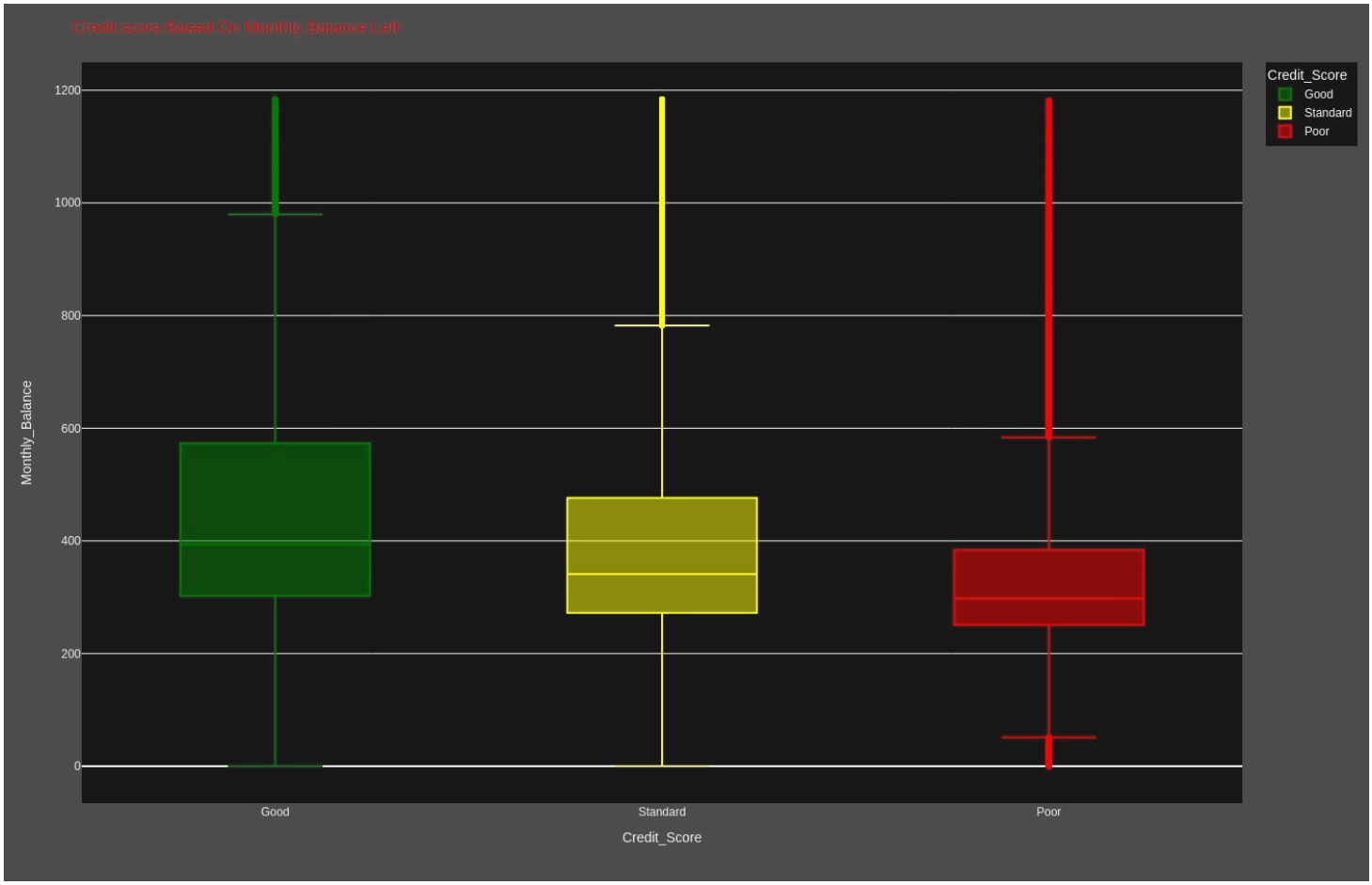
[24]:
```
fig=px.box(data,
                x='Credit_Score',
                y='Monthly_Balance',
                color='Credit_Score',
                color_discrete_map={'Poor':'red',
                                    'Standard':'yellow',
                                    'Good':'green'},
                )
fig.update_traces(quartilemethod='exclusive')
fig.update_layout(
    plot_bgcolor='rgba(0,0,0,0.7)', # sets the plot background color to a dark␣
  gray
    paper_bgcolor='rgba(0,0,0,0.7)', # sets the paper background color to a␣
  dark gray
    font_color='white',
    title={'text':'Credit score Based On Monthly Balance Left','font_color':
  'red'},
    width=1400,
    height=900
)
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.

4.0.1 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 Based On Amount Invested Monthly
Credit_Score
Good
Standard
Poor
Amount_invested_monthly
450
400
350
300
250
200
150
100
50
0
Good
Standard
Poor
Credit_Score


Credit score Based On Monthly Balance Left
Credit_Score
Good
Standard
Poor
Monthly_Balance
1200
1000
800
600
400
200
0
Good
Standard
Poor
Credit_Score

credit score classification:

[25]:
```
data['Credit_Mix']=data['Credit_Mix'].map({"Standard":1,"Good":2,"Bad":0})
```

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

[26]:
```
from sklearn.model_selection import train_test_split
```

- Making Dependent and Independent Feature

[27]:
```
# Independent Feature
x = np.array(data[["Annual_Income", "Monthly_Inhand_Salary",
                   "Num_Bank_Accounts", "Num_Credit_Card",
                   "Interest_Rate", "Num_of_Loan",
                   "Delay_from_due_date", "Num_of_Delayed_Payment",
                   "Credit_Mix", "Outstanding_Debt",
                   "Credit_History_Age", "Monthly_Balance"]]
)
  # dependent Features
y = np.array(data["Credit_Score"])
```

[28]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y,
                                                    test_size=0.33,
                                                    random_state=42)
```

[29]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(x_train, y_train)
```

[29]:
```
RandomForestClassifier()
```

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

[30]:
```
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 the person: "))
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, k, l]])
print("Predicted Credit Score = ", model.predict(features))
```

Credit Score Prediction :
Annual Income: 19114.12
Monthly Inhand Salary: 1824.6
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']

-

5 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.

6 Reference

- DataSet Link (Click Me
- Aman Kharwal (medium.com)


```
<h2 style='padding: 20px;
                color:red;
                text-align:center;'>
        END OF THE PROJECT !
</h2>
</div>


<h2 style='padding: 20px;
                color:GREEN;
```

```
                text-align:center;'>
          THANK YOU !
        </h2>
</div>
```