In [3]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
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(r'C:\Users\mamta\Desktop\DESKTOP MAMTA\Bsc Projects\ml project
print(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

   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

   Outstanding_Debt  Credit_Utilization_Ratio Credit_History_Age  \
0            809.98                  26.82262              265.0
1            809.98                  31.94496              266.0

   Payment_of_Min_Amount  Total_EMI_per_month  Amount_invested_monthly  \
0                     No            49.574949                 21.46538
1                     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 rows x 28 columns]
```

In [4]:

```python
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
```

In [5]:

```python
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
```

In [6]:

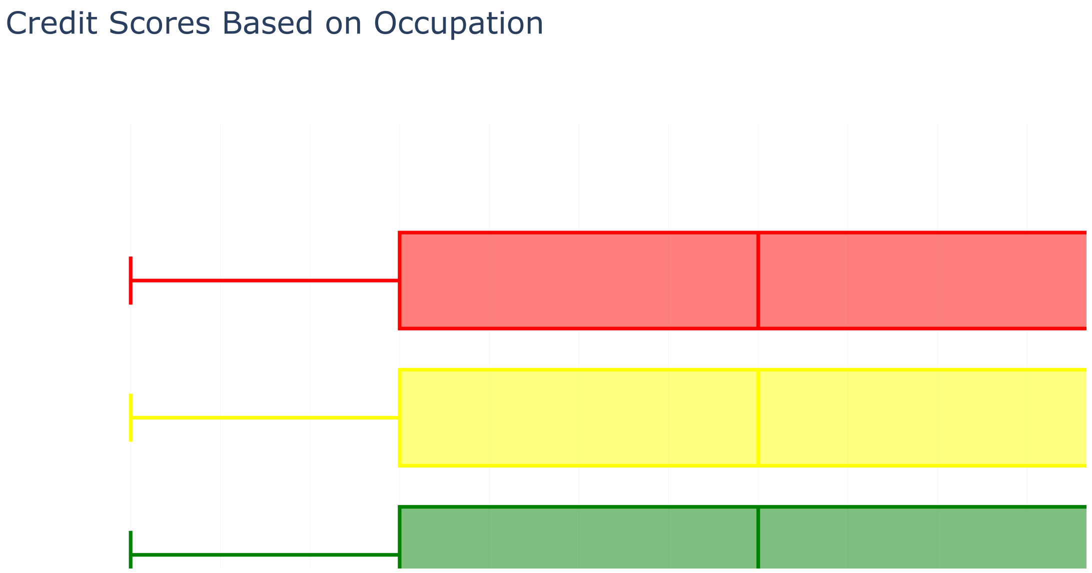
```python
#credit score column values
data["Credit_Score"].value_counts()
```

Out[6]:

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

In [7]:

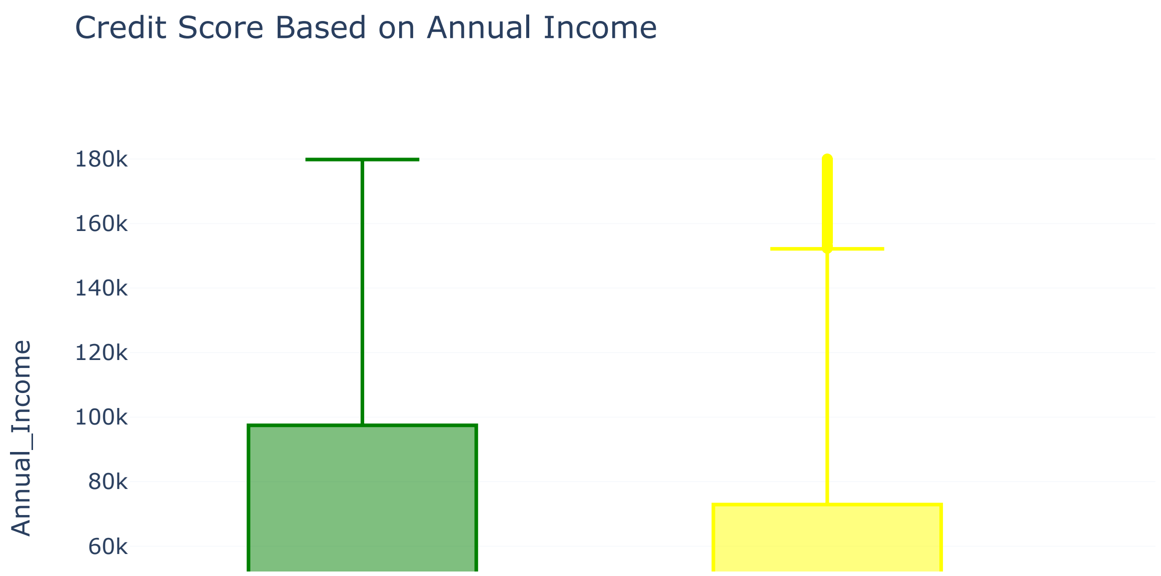
```
#Data Exploration
plt.figure(figsize = (5,4))
fig = px.box(data,x = "Occupation",color = "Credit_Score",title = "Credit Scores
             color_discrete_map = {'Poor':'red','Standard':'yellow','Good':'green'
fig.show()
```



```
<Figure size 500x400 with 0 Axes>
```

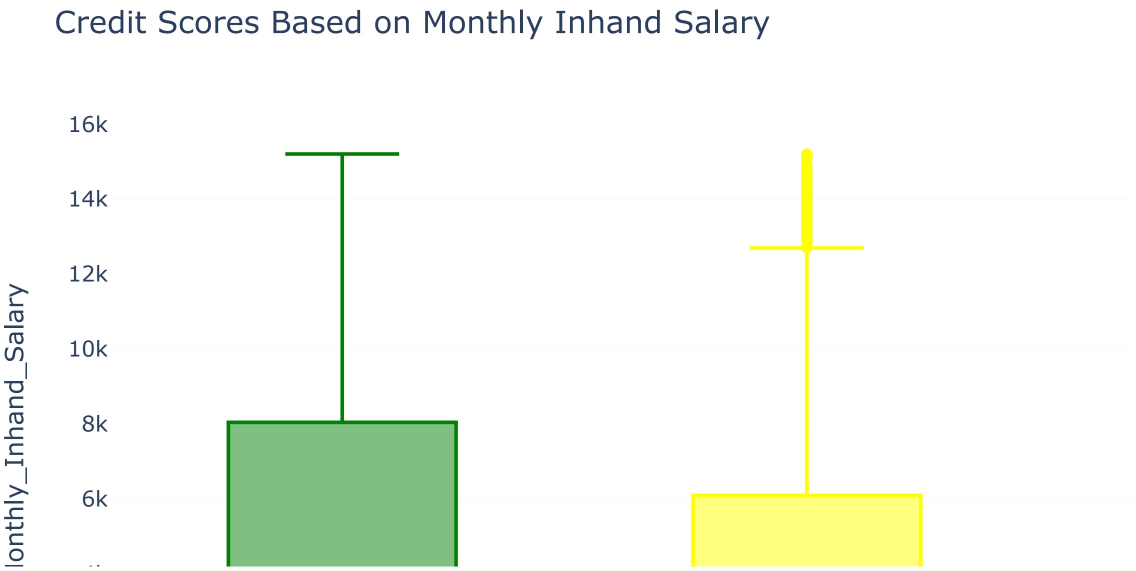
In [8]:

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



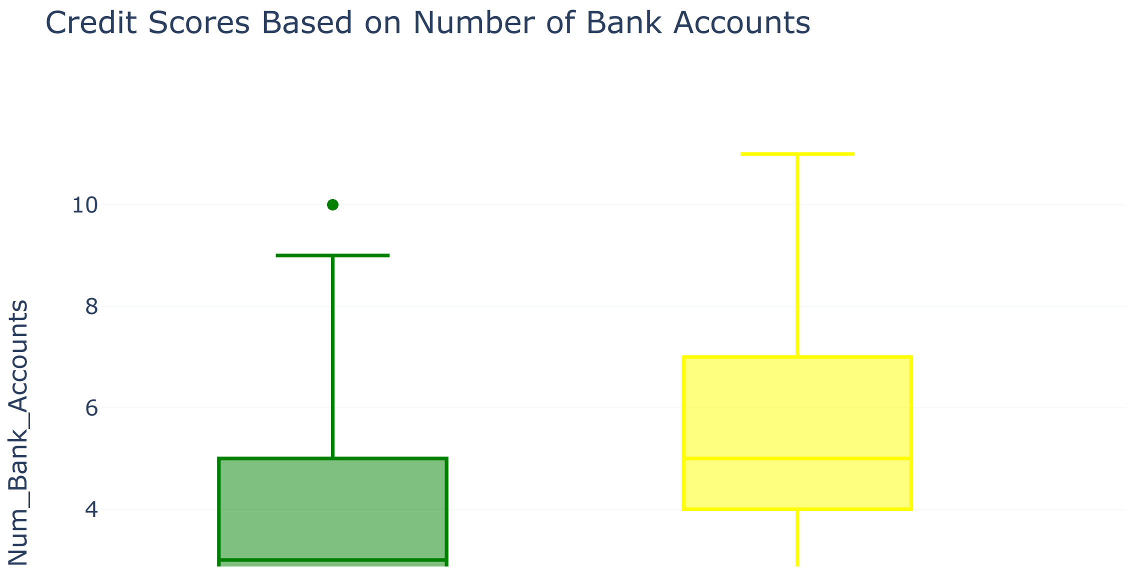
In [9]:

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



In [10]:

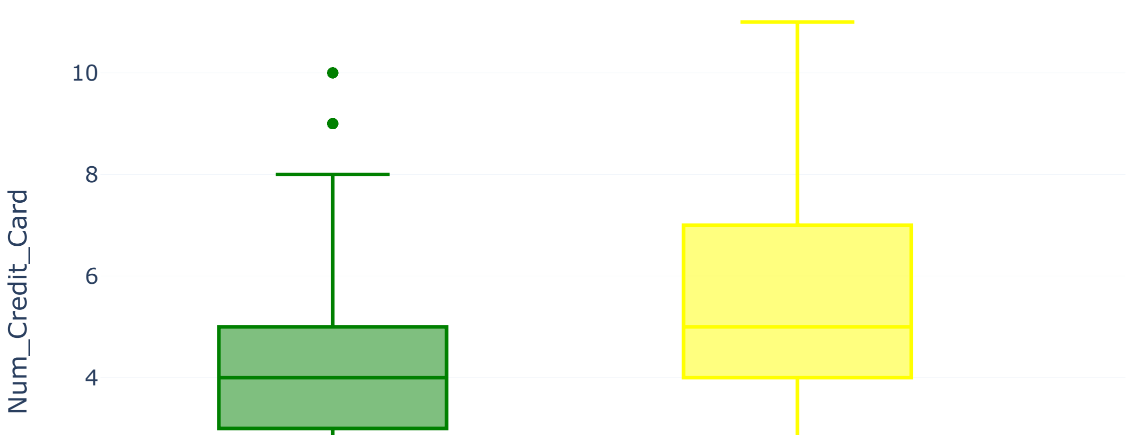
```
fig = px.box(data,x = "Credit_Score", y = "Num_Bank_Accounts",color = "Credit_Sco
             title = "Credit Scores Based on Number of Bank Accounts",
             color_discrete_map = {'Poor':'red','Standard':'yellow','Good':'green'
fig.update_traces(quartilemethod="exclusive")
fig.show()
```



Credit Scores Based on Number of Bank Accounts

In [11]:

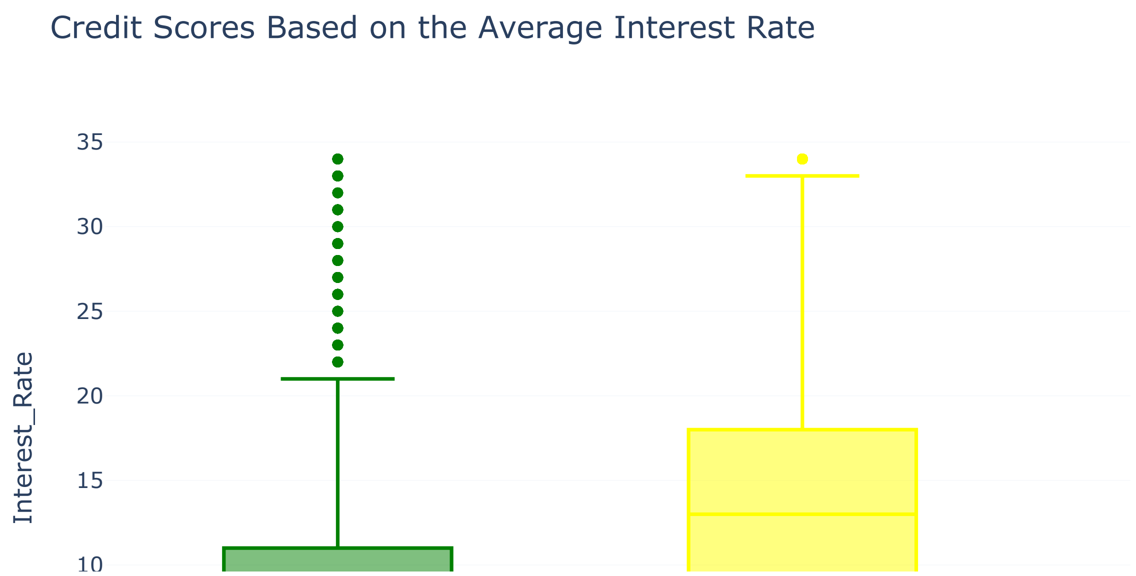
```
fig = px.box(data,x = "Credit_Score",y = "Num_Credit_Card",color = "Credit_Score"
             title = "Credit Scores Based on Number of Credit Cards",
             color_discrete_map = {'Poor':'red','Standard':'yellow','Good':'green'
fig.update_traces(quartilemethod = "exclusive")
fig.show()
```



Credit Scores Based on Number of Credit Cards

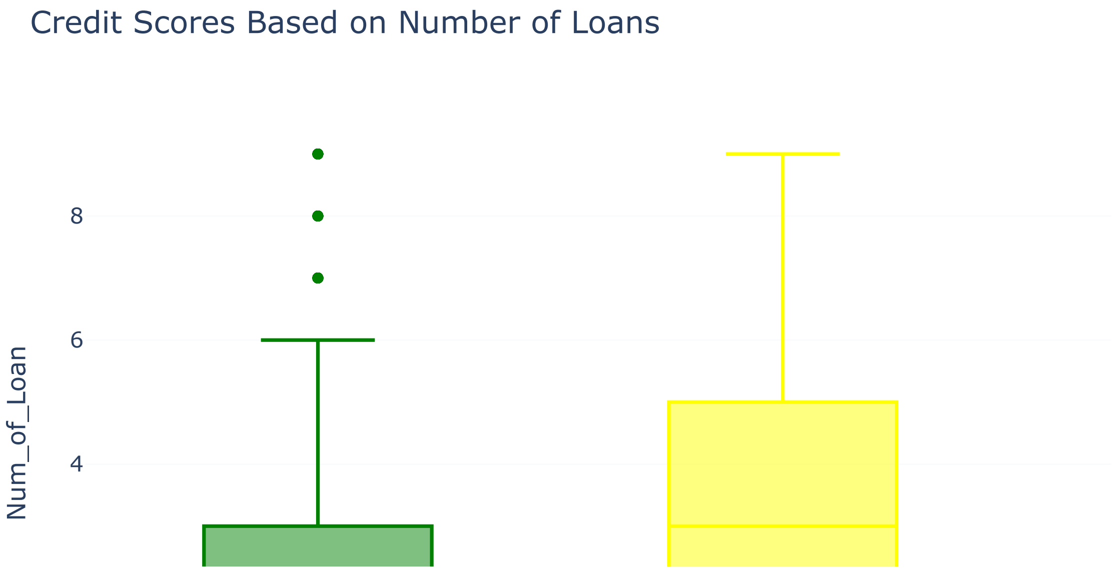
In [12]:

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


Credit Scores Based on the Average Interest Rate

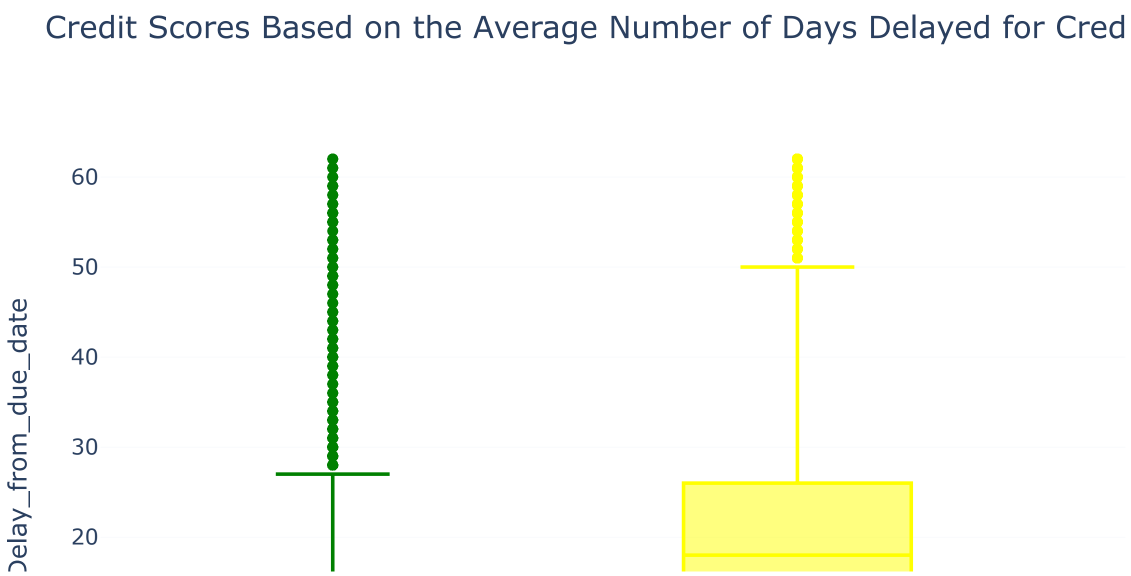
In [13]:

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



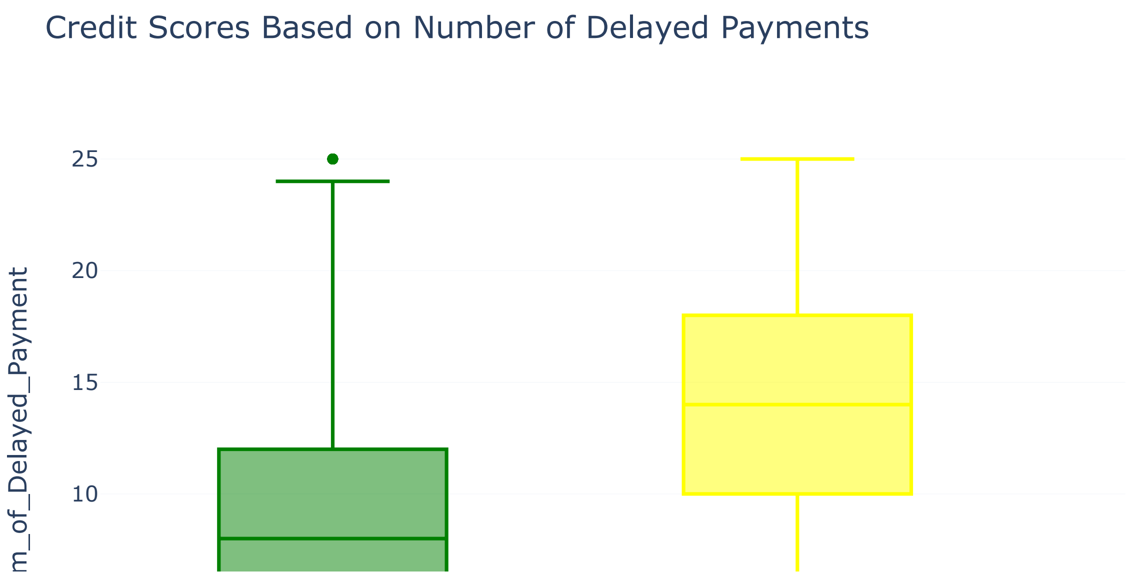
In [14]:

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



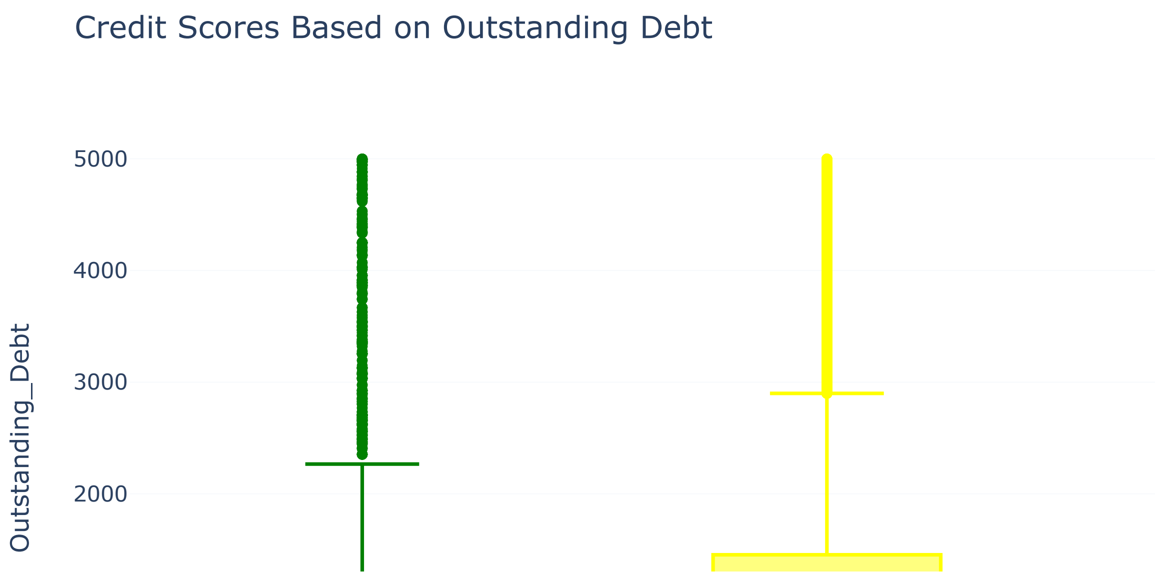
In [15]:

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



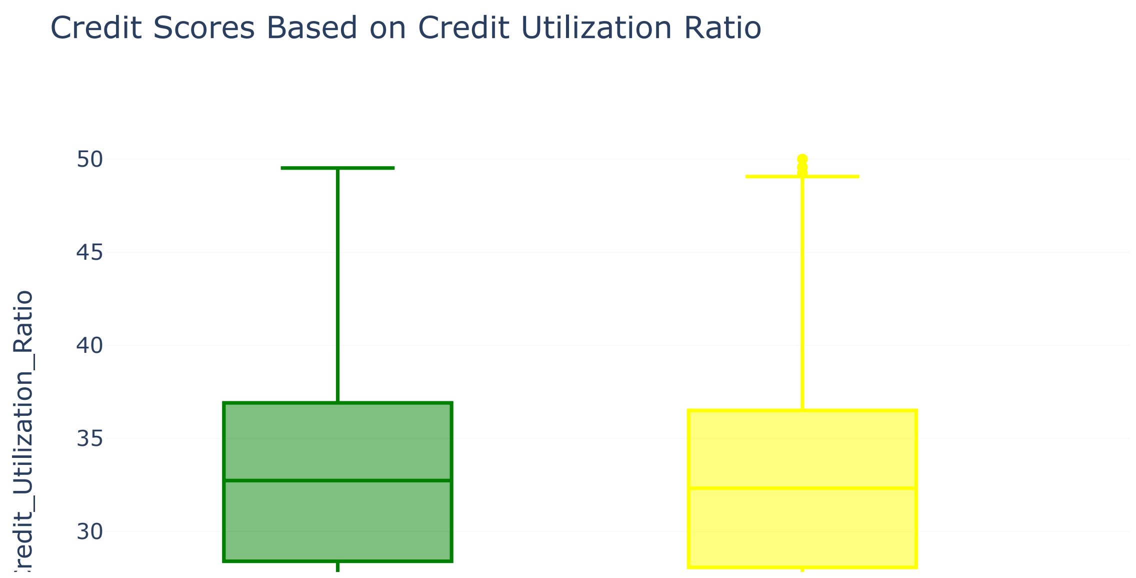
In [16]:

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



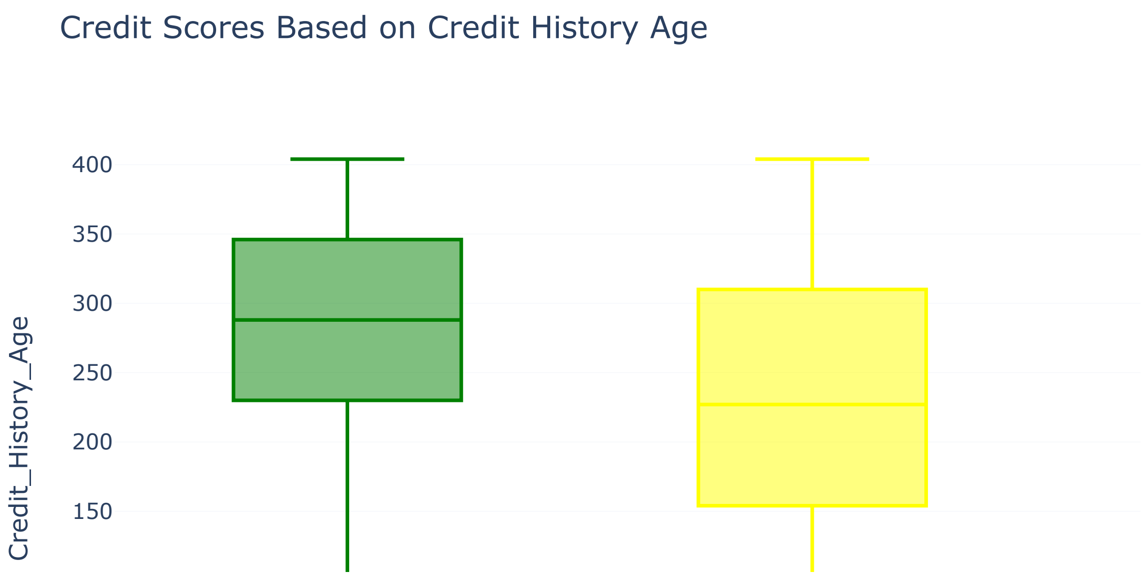
In [17]:

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



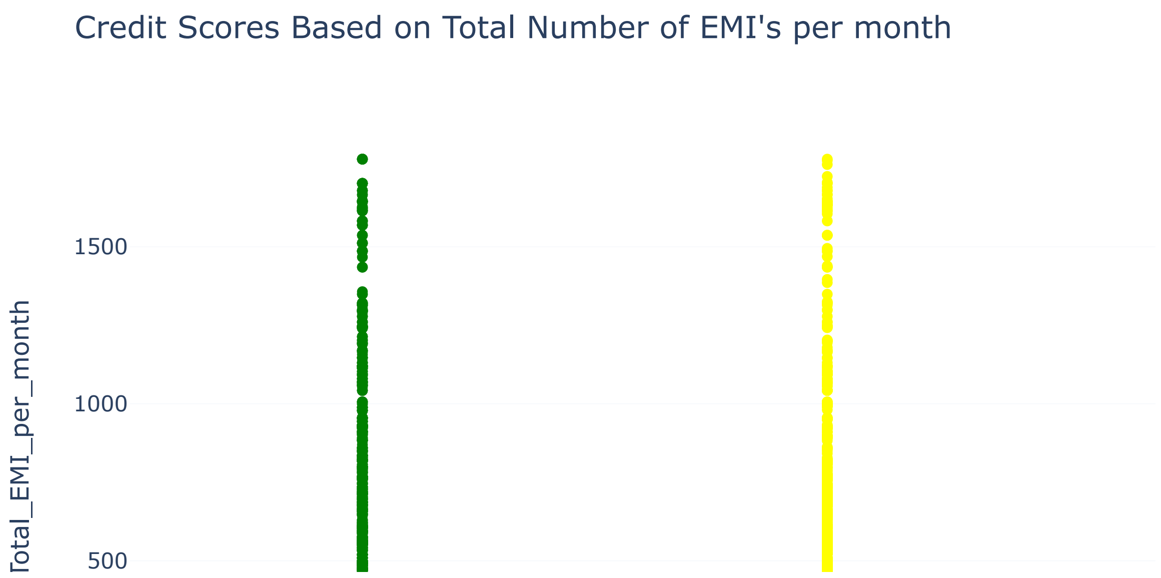
In [18]:

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



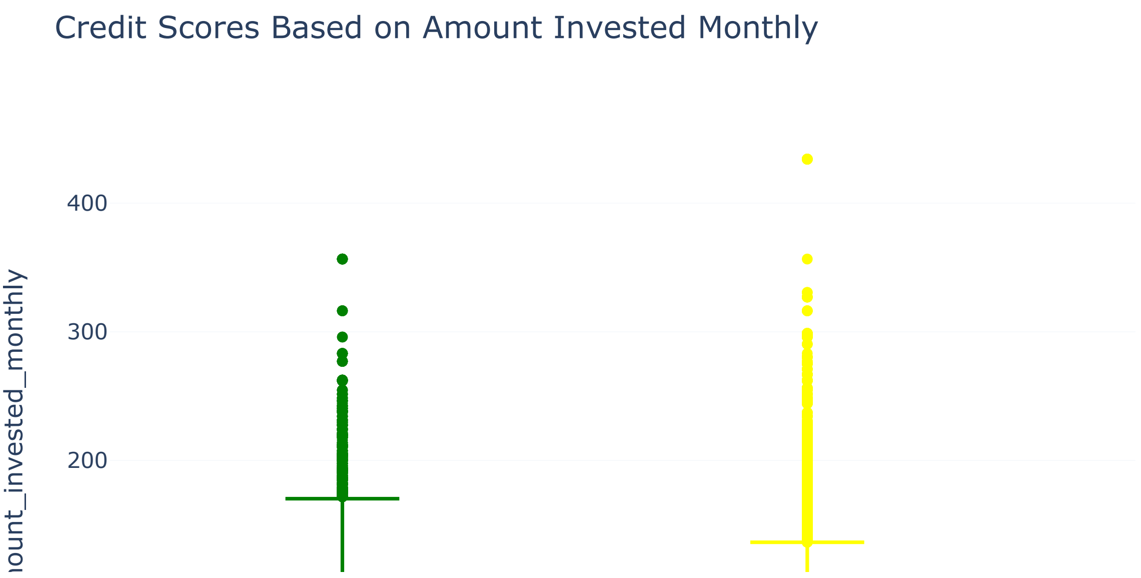
In [19]:

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



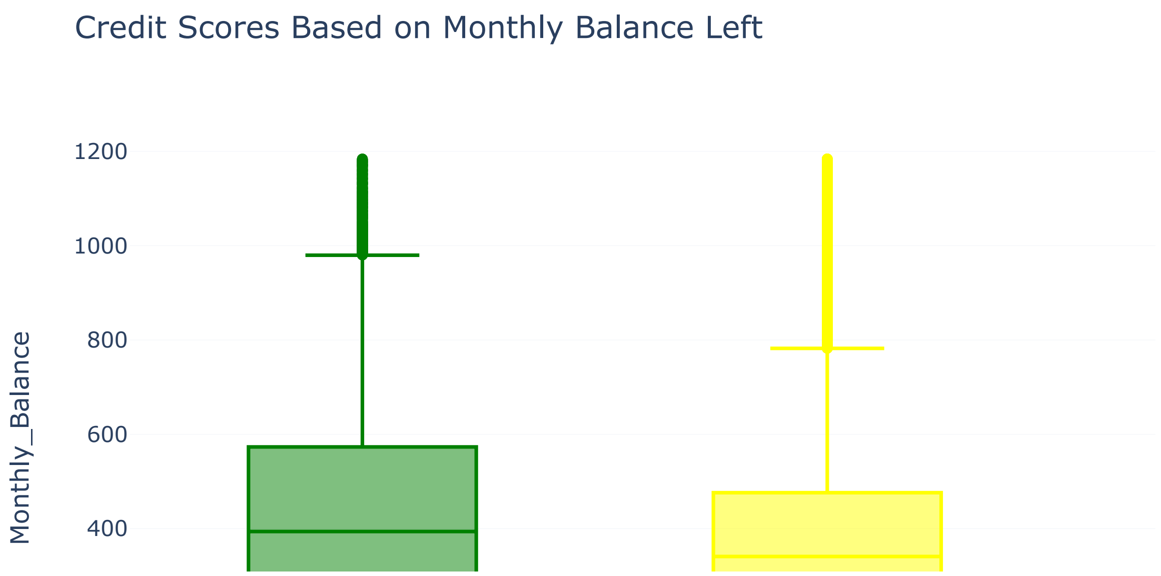
In [20]:

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



In [21]:

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



In [22]:

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

In [23]:

```python
from sklearn.model_selection import train_test_split
x = np.array(data[["Annual_Income","Monthly_Inhand_Salary","Num_Bank_Accounts",
                   "Num_Credit_Card","Interest_Rate","Num_of_Loan","Delay_from_due
                   "Num_of_Delayed_Payment","Credit_Mix","Outstanding_Debt","Credi
                   "Monthly_Balance"]])
y = np.array(data[["Credit_Score"]])
```

In [24]:

```python
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size = 0.33,random_state=42
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(xtrain,ytrain)
```

```
C:\Users\mamta\AppData\Local\Temp\ipykernel_19276\790351916.py:4: DataConversionWarn
ing:

A column-vector y was passed when a 1d array was expected. Please change the shape o
f y to (n_samples,), for example using ravel().
```

Out[24]:

```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [25]:

```python
def 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: "))
    return [a,b,c,d,e,f,g,h,i,j,k,l]

features = np.array(credit_score_prediction())
print("Predicted Credit Score = ", model.predict(features))
```

```
Annual Income: 2500000
Monthly Inhand Salary: 218000
Number of Bank Accounts: 2
Number of Credit Cards: 1
Interest rate: 12
Number of Loans: 2
Average number of days delayed by the person: 5
Number of delayed payments: 1
Credit Mix (Bad: 0, Standard: 1, Good: 3): 3
Outstanding Debt: 1200
Credit History Age: 2
Monthly Balance: 500
Predicted Credit Score =  ['Good']
```

In [ ]:

In [ ]: