In [478…

```
# Accessing necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from dmba import plotDecisionTree
from sklearn.naive_bayes import MultinomialNB
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
from sklearn.preprocessing import MinMaxScaler
```

In [479…

```
# Read the main table
CRSData = pd.read_excel('train.xlsx', sheet_name='train_5K')
CRSData.round(2).head(5)  # Display the first few rows of the dataframe
```

Out[479…

	ID	Customer_ID	Age	SSN	Occupation	AnnualIncome	Monthly_Inhand_Salary	N
0	0x1602	CUS_0xd40	23	821-00-0265	Scientist	19114.12	1824.84	
1	0x1603	CUS_0xd40	23	821-00-0265	Scientist	19114.12	1824.84	
2	0x1604	CUS_0xd40	23	821-00-0265	Scientist	19114.12	1824.84	
3	0x1605	CUS_0xd40	23	821-00-0265	Scientist	19114.12	1824.84	
4	0x1606	CUS_0xd40	23	821-00-0265	Scientist	19114.12	1824.84	

5 rows × 27 columns

In [480…

```
CRSData.isnull().sum()
```

Out[480…

```
ID                           0
Customer_ID                  0
Age                          0
SSN                          0
Occupation                   0
AnnualIncome                 0
Monthly_Inhand_Salary        0
Num_Bank_Accounts            0
Num_Credit_Card              0
Interest_Rate                0
NumofLoan                    0
Type_of_Loan                 0
Month                        0
Delay_from_due_date          0
Num_of_Delayed_Payment     347
ChangedCreditLimit         109
Num_Credit_Inquiries       111
Credit_Mix                   0
OutstandingDebt              0
Credit_Utilization_Ratio     0
Credit_History_Age         474
Payment_of_Min_Amount        0
Total_EMI_per_month          0
Amount_invested_monthly    213
Payment_Behaviour          394
Monthly_Balance             66
Credit_Score                 0
dtype: int64
```

In [481…

```
# Dropping irrelevant columns
CRSData.drop(['Month' , 'Type_of_Loan', 'Credit_History_Age', 'SSN','Credit_Mix'],
```

In [482…

```
#Treating missing values
CRSData['Num_of_Delayed_Payment'].fillna(CRSData['Num_of_Delayed_Payment'].median()
CRSData['ChangedCreditLimit'].fillna(CRSData['ChangedCreditLimit'].median(), inplac
CRSData['Num_Credit_Inquiries'].fillna(CRSData['Num_Credit_Inquiries'].median(), in
CRSData['Amount_invested_monthly'].fillna(CRSData['Amount_invested_monthly'].mean()
CRSData['Monthly_Balance'].fillna(CRSData['Monthly_Balance'].median(), inplace=True
```

In [483…

```
#Removing missing values from payment behaviour
CRSclean_df = CRSData.dropna(subset=['Payment_Behaviour'])
CRSclean_df.isnull().sum()
```

Out[483…
```
ID                          0
Customer_ID                 0
Age                         0
Occupation                  0
AnnualIncome                0
Monthly_Inhand_Salary       0
Num_Bank_Accounts           0
Num_Credit_Card             0
Interest_Rate               0
NumofLoan                   0
Delay_from_due_date         0
Num_of_Delayed_Payment      0
ChangedCreditLimit          0
Num_Credit_Inquiries        0
OutstandingDebt             0
Credit_Utilization_Ratio    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 [484…
```
CRSclean_df.dtypes
```

Out[484…
```
ID                           object
Customer_ID                  object
Age                           int64
Occupation                   object
AnnualIncome                float64
Monthly_Inhand_Salary       float64
Num_Bank_Accounts             int64
Num_Credit_Card               int64
Interest_Rate                 int64
NumofLoan                     int64
Delay_from_due_date           int64
Num_of_Delayed_Payment      float64
ChangedCreditLimit          float64
Num_Credit_Inquiries        float64
OutstandingDebt             float64
Credit_Utilization_Ratio    float64
Payment_of_Min_Amount        object
Total_EMI_per_month         float64
Amount_invested_monthly     float64
Payment_Behaviour            object
Monthly_Balance             float64
Credit_Score                 object
dtype: object
```

In [485…
```
CRSclean_df.describe().round(2)
```

Out[485…

	Age	AnnualIncome	Monthly_Inhand_Salary	Num_Bank_Accounts	Num_Credit_Ca
count	4606.00	4606.00	4606.00	4606.00	4606
mean	32.91	234643.88	3605.42	5.13	5
std	11.10	1638638.01	3305.15	2.38	1
min	0.00	7103.04	0.00	0.00	0
25%	24.00	19795.52	1223.39	3.00	4
50%	33.00	37131.02	2693.35	6.00	5
75%	42.00	72559.36	5242.95	7.00	6
max	76.00	20976455.00	14710.53	9.00	9

In [486…

```
#Removing 'NM' values in the column 'Payment of Min Amount'
CRSclean_df = CRSclean_df[CRSclean_df['Payment_of_Min_Amount'] != 'NM']
```

In [487…

```
numeric_cols = CRSclean_df.select_dtypes(exclude = "object").columns
cat_cols = CRSclean_df.select_dtypes(include = "object").columns
print(numeric_cols)
print(cat_cols)
```

```
Index(['Age', 'AnnualIncome', 'Monthly_Inhand_Salary', 'Num_Bank_Accounts',
       'Num_Credit_Card', 'Interest_Rate', 'NumofLoan', 'Delay_from_due_date',
       'Num_of_Delayed_Payment', 'ChangedCreditLimit', 'Num_Credit_Inquiries',
       'OutstandingDebt', 'Credit_Utilization_Ratio', 'Total_EMI_per_month',
       'Amount_invested_monthly', 'Monthly_Balance'],
      dtype='object')
Index(['ID', 'Customer_ID', 'Occupation', 'Payment_of_Min_Amount',
       'Payment_Behaviour', 'Credit_Score'],
      dtype='object')
```

In [488…

```
#Checking Multicollinearity
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif_df = CRSclean_df[numeric_cols]
vif_data = pd.DataFrame({
    "feature": vif_df.columns,
    "VIF": [variance_inflation_factor(vif_df.values, i) for i in range(len(vif_df.c
})
print(vif_data.head(17).round(2))
```

	feature	VIF
0	Age	8.67
1	AnnualIncome	1.04
2	Monthly_Inhand_Salary	3.37
3	Num_Bank_Accounts	8.72
4	Num_Credit_Card	10.39
5	Interest_Rate	8.05
6	NumofLoan	5.26
7	Delay_from_due_date	5.95
8	Num_of_Delayed_Payment	1.03
9	ChangedCreditLimit	4.54
10	Num_Credit_Inquiries	1.02
11	OutstandingDebt	5.77
12	Credit_Utilization_Ratio	21.26
13	Total_EMI_per_month	1.03
14	Amount_invested_monthly	1.11
15	Monthly_Balance	8.30

In [489…

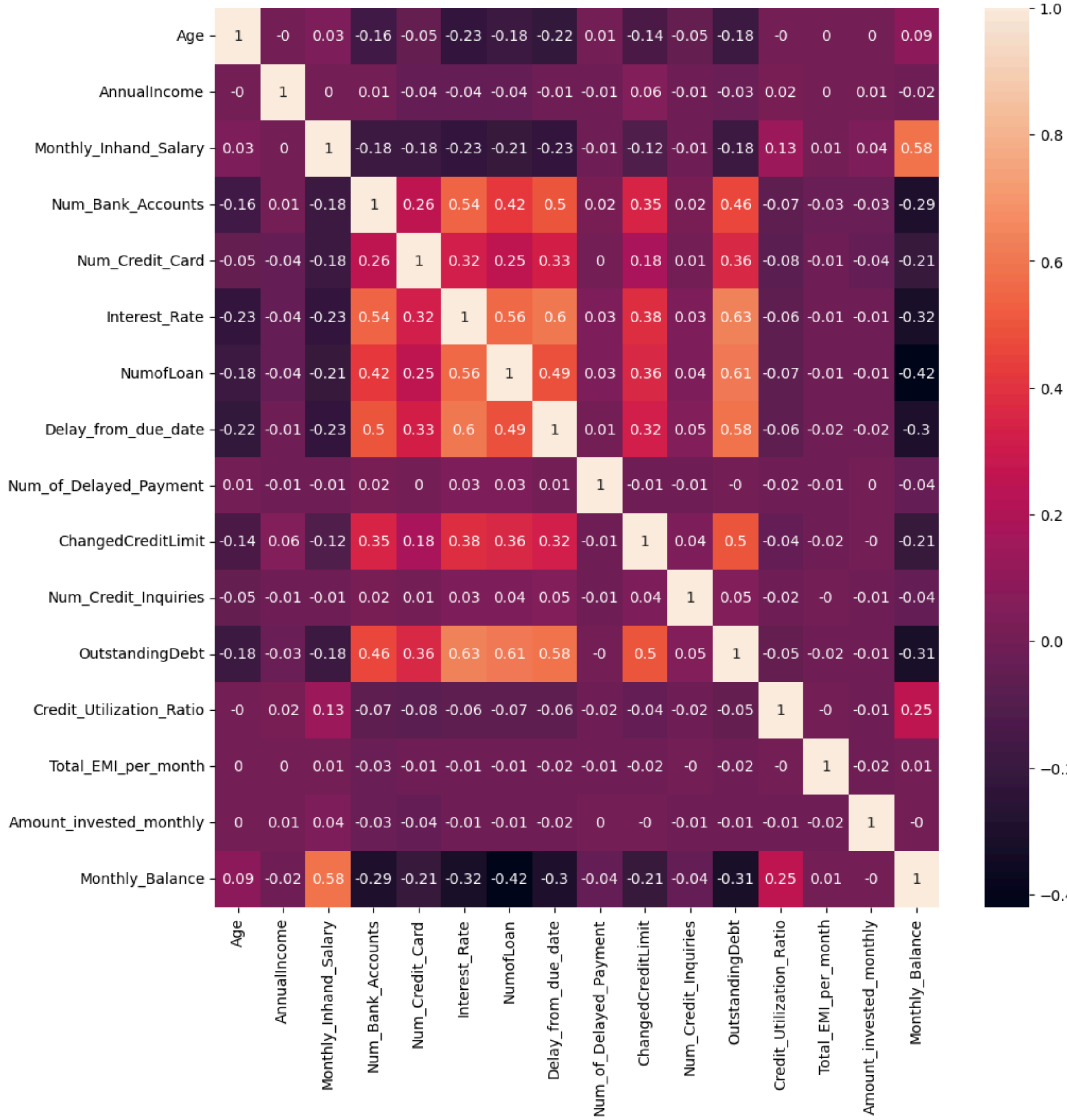
```
#Shows few variables as High multicollinearity, indicating that the predictor is hi
#We will use feature selection for this at the later steps
```

In [490…

```
plt.figure(figsize= (11,11))
sns.heatmap(CRSclean_df[numeric_cols].corr().round(2),annot=True)
```

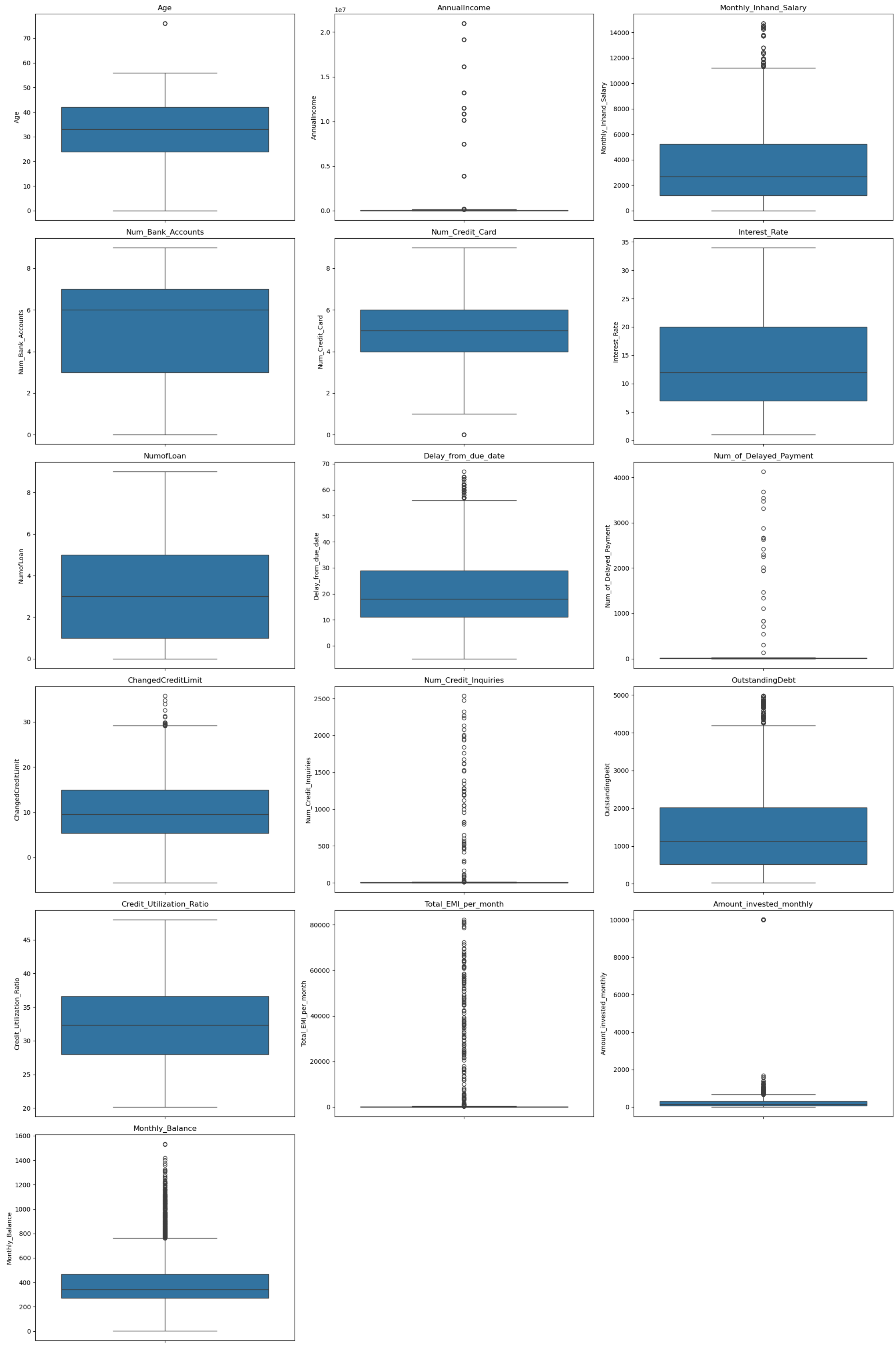
Out[490…

```
<Axes: >
```



In [491…

```
#Visualising boxplot
# Set the number of rows and columns for the subplots grid
num_cols = 3  # Number of columns in the subplot grid
num_rows = int(np.ceil(len(numeric_cols) / num_cols))  # Calculate the number of ro
# Set the figure size for better visibility
plt.figure(figsize=(20, num_rows * 5))
# Create a boxplot for each numerical column
for i, col in enumerate(numeric_cols):
    plt.subplot(num_rows, num_cols, i + 1)
    sns.boxplot(y=CRSclean_df[col])
    plt.title(col)
    plt.xlabel('')
# Adjust layout to prevent overlap
plt.tight_layout()
# Show the plot
plt.show()
```


Age
AnnualIncome
1e7
Monthly_Inhand_Salary
Num_Bank_Accounts
Num_Credit_Card
Interest_Rate
NumofLoan
Delay_from_due_date
Num_of_Delayed_Payment
ChangedCreditLimit
Num_Credit_Inquiries
OutstandingDebt
Credit_Utilization_Ratio
Total_EMI_per_month
Amount_invested_monthly
Monthly_Balance

In [492…

```
#Replacing outlier with median

CRS_o_df = CRSclean_df[numeric_cols].copy()

for col in numeric_cols:
    # Calculate the 0.05th and 99.95th percentiles
    Q1 = np.percentile(CRS_o_df[col], 0.05, interpolation='midpoint')
    Q3 = np.percentile(CRS_o_df[col], 99.95, interpolation='midpoint')
    median = CRS_o_df[col].median()
# Replace outliers with the median
CRS_o_df[col] = np.where((CRS_o_df[col] < Q1) | (CRS_o_df[col] > Q3), median,CRS_o_
CRS_o_df = CRS_o_df.round(4)

# Display the first few rows of the cleaned DataFrame
print(CRS_o_df)
```

```
      Age  AnnualIncome  Monthly_Inhand_Salary  Num_Bank_Accounts  \
0      23      19114.12              1824.8433                  3   
1      23      19114.12              1824.8433                  3   
2      23      19114.12              1824.8433                  3   
3      23      19114.12              1824.8433                  3   
4      23      19114.12              1824.8433                  3   
...   ...           ...                    ...                ...   
4992   20      77519.04              6184.9200                  6   
4994   20      77519.04              6184.9200                  6   
4996   20      77519.04              6184.9200                  6   
4998   20      77519.04              6184.9200                  6   
4999   20      77519.04              6184.9200                  6   

      Num_Credit_Card  Interest_Rate  NumofLoan  Delay_from_due_date  \
0                   4              3          4                    3   
1                   4              3          4                   -1   
2                   4              3          4                    3   
3                   4              3          4                    5   
4                   4              3          4                    6   
...               ...            ...        ...                  ...   
4992                6             23          7                   32   
4994                6             23          7                   29   
4996                6             23          7                   31   
4998                6             23          7                   31   
4999                6             23          7                   31   

      Num_of_Delayed_Payment  ChangedCreditLimit  Num_Credit_Inquiries  \
0                        7.0               11.27                   4.0   
1                       14.0               11.27                   4.0   
2                        7.0                9.54                   4.0   
3                        4.0                6.27                   4.0   
4                       14.0               11.27                   4.0   
...                      ...                 ...                   ...   
4992                    16.0               16.57                   6.0   
4994                    14.0               16.57                   6.0   
4996                    17.0               16.57                   6.0   
4998                    16.0               16.57                   6.0   
4999                    14.0               16.57                   6.0   

      OutstandingDebt  Credit_Utilization_Ratio  Total_EMI_per_month  \
0              809.98                   26.8226              49.5749   
1              809.98                   31.9450              49.5749   
2              809.98                   28.6094              49.5749   
3              809.98                   31.3779              49.5749   
4              809.98                   24.7973              49.5749   
...               ...                       ...                  ...   
4992          3343.32                   35.1656             360.6813   
4994          3343.32                   22.7776             360.6813   
4996          3343.32                   23.2345             360.6813   
4998          3343.32                   26.3352             360.6813   
4999          3343.32                   37.6528             360.6813   

      Amount_invested_monthly  Monthly_Balance  
0                     80.4153         312.4941  
1                    118.2802         284.6292  
2                     81.6995         331.2099  
```

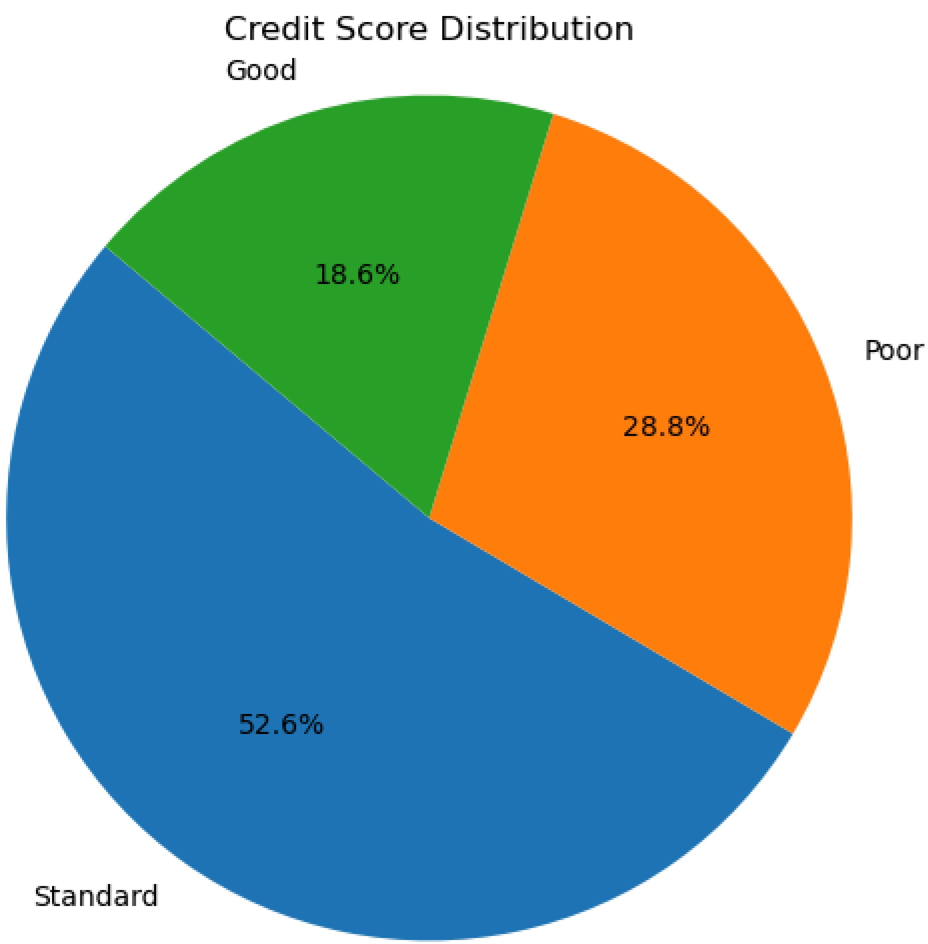
```
3                    199.4581          223.4513
4                     41.4202          341.4892
...                       ...               ...
4992                 287.1395          250.6712
4994                 615.9845          340.8869
4996                  68.5376          429.2731
4998                 147.3195          350.4912
4999                 500.8134           36.9973

[4058 rows x 16 columns]
```

In [493…

```
#Visualisation
```

In [494…

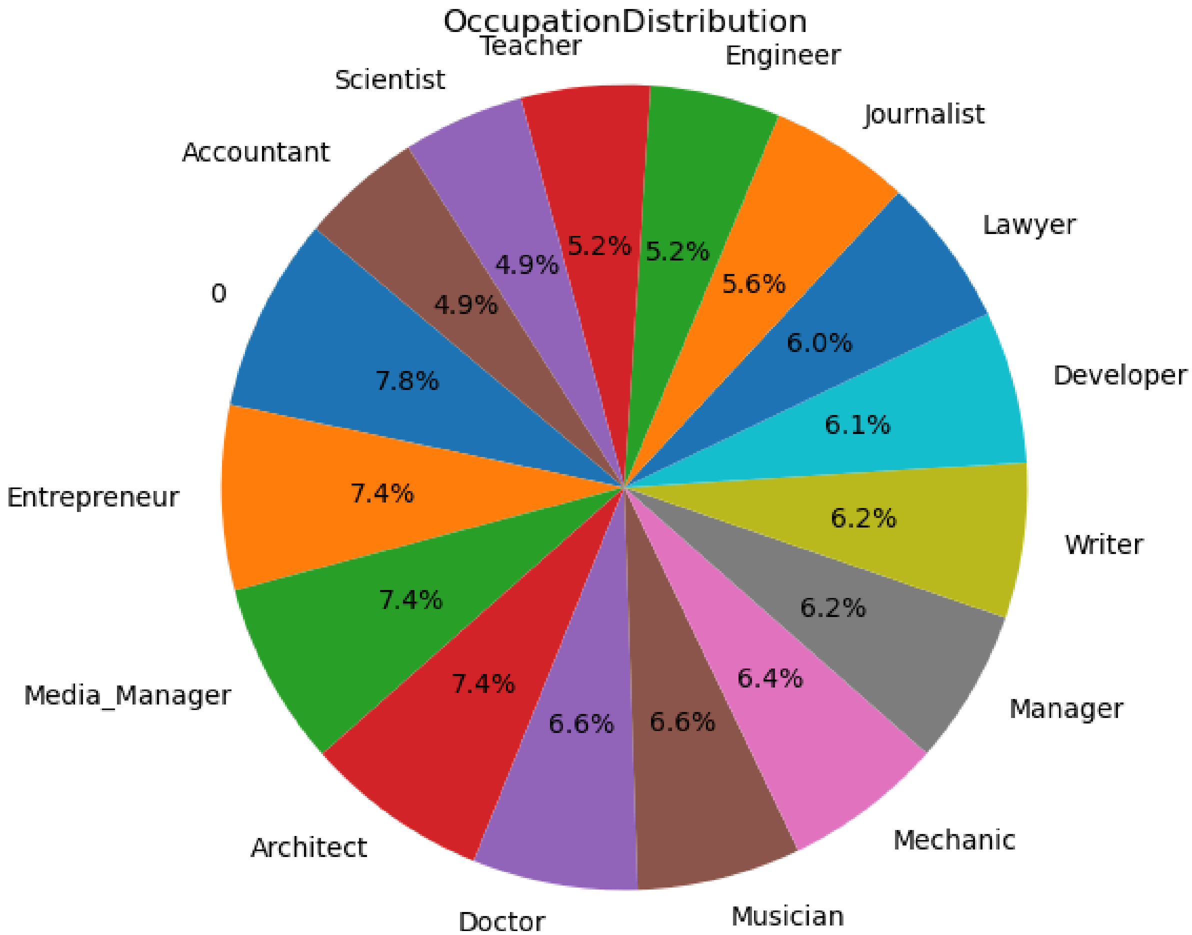
```
#Proportion of credit score
credit_score_counts =CRSclean_df['Credit_Score'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(credit_score_counts, labels=credit_score_counts.index, autopct='%1.1f%%', s
plt.title('Credit Score Distribution')
plt.axis('equal')
plt.show()
```



In [495…

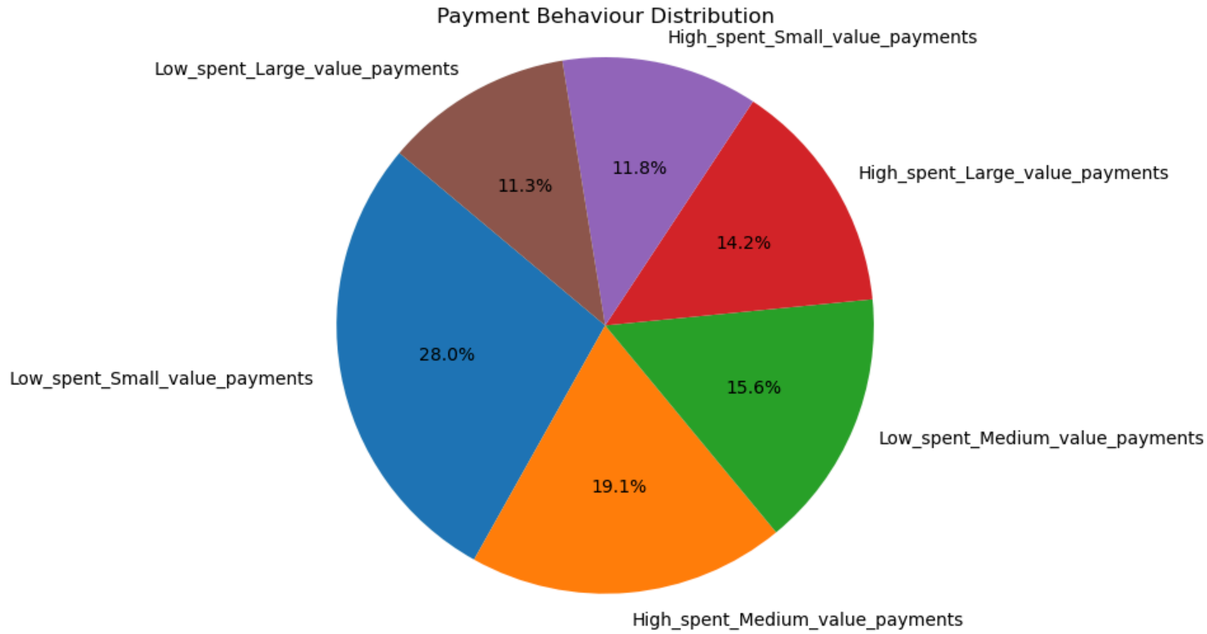
```
#Proportion of occupation
Occupation_counts =CRSclean_df['Occupation'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(Occupation_counts, labels=Occupation_counts.index, autopct='%1.1f%%', start
```

```
plt.title('OccupationDistribution')
plt.axis('equal')
plt.show()
```



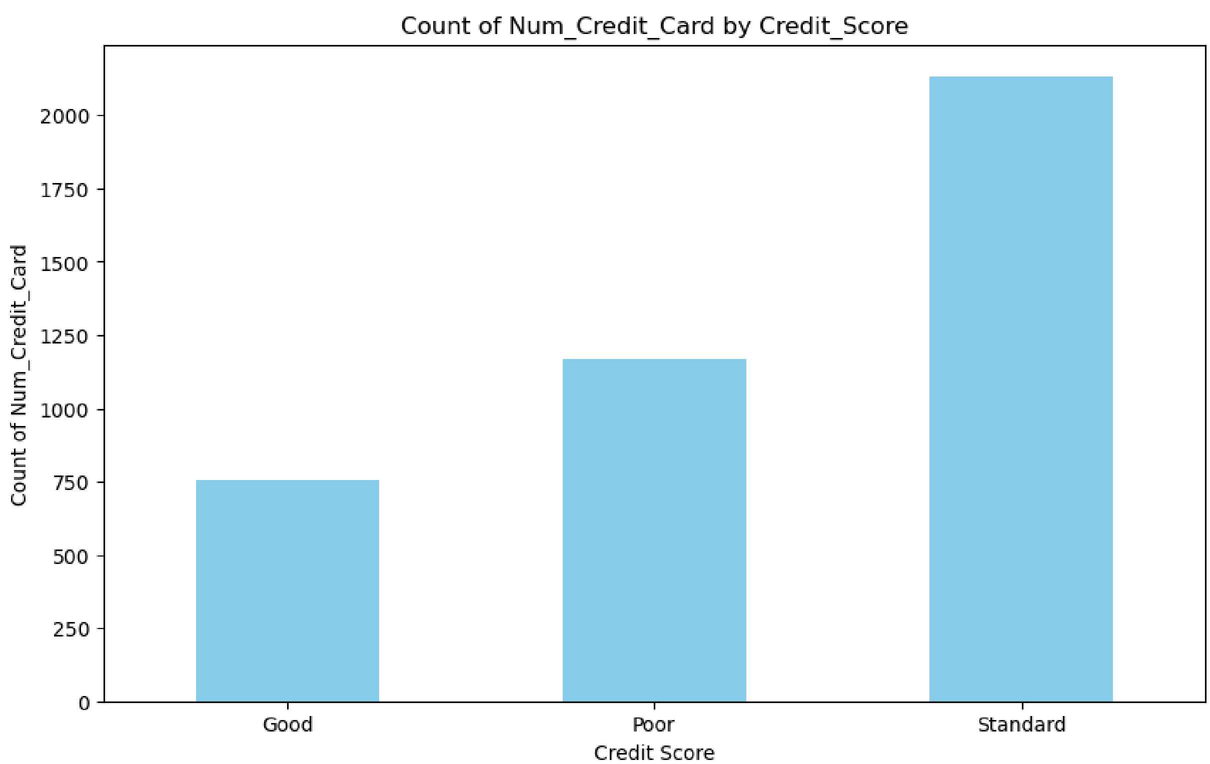
In [496…

```
#Proportion of Payment Bheaviour
PB_counts =CRSclean_df['Payment_Behaviour'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(PB_counts, labels=PB_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Payment Behaviour Distribution')
plt.axis('equal')
plt.show()
```



In [497…

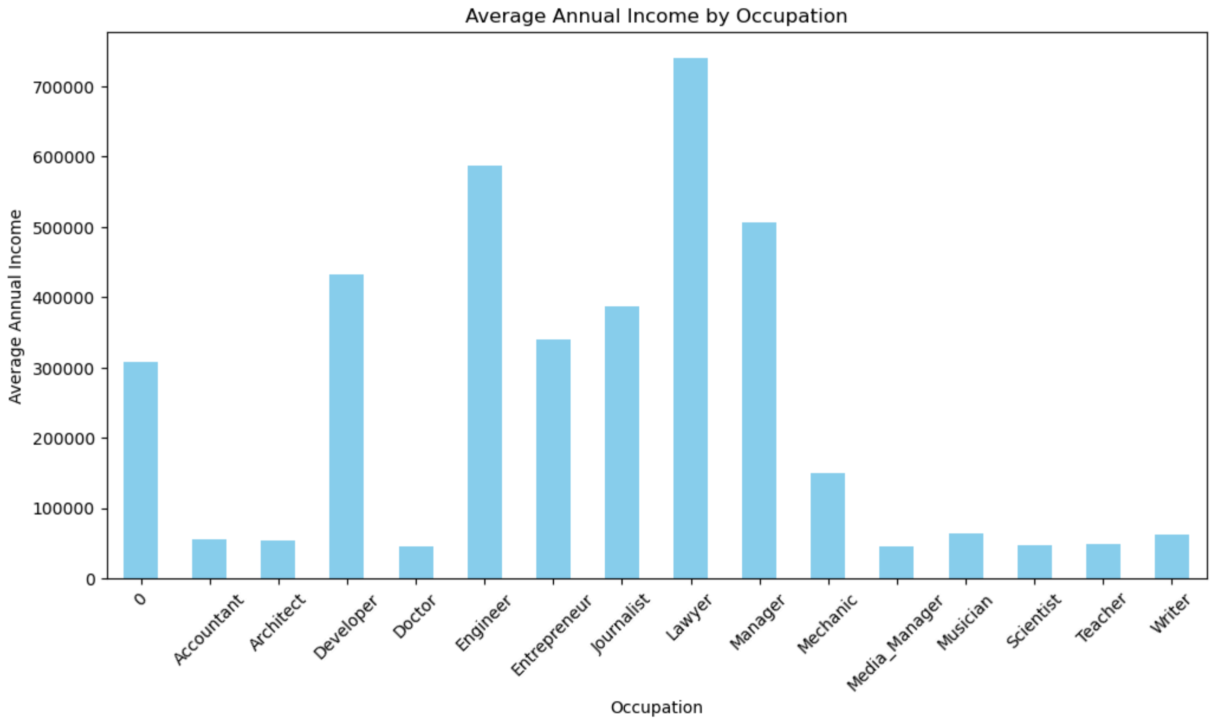
```
#Number of credit card by credit score
credit_card_counts = CRSclean_df.groupby('Credit_Score')['Num_Credit_Card'].count()
plt.figure(figsize=(10, 6))
credit_card_counts.plot(kind='bar', color='skyblue')
plt.title('Count of Num_Credit_Card by Credit_Score')
plt.xlabel('Credit Score')
plt.ylabel('Count of Num_Credit_Card')
plt.xticks(rotation=0)
plt.show()
```



In [498…

```
average_income_by_occupation = CRSclean_df.groupby('Occupation')['AnnualIncome'].me
plt.figure(figsize=(12, 6))
```

```
average_income_by_occupation.plot(kind='bar', color='skyblue')
plt.title('Average Annual Income by Occupation')
plt.xlabel('Occupation')
plt.ylabel('Average Annual Income')
plt.xticks(rotation=45)
plt.show()
```

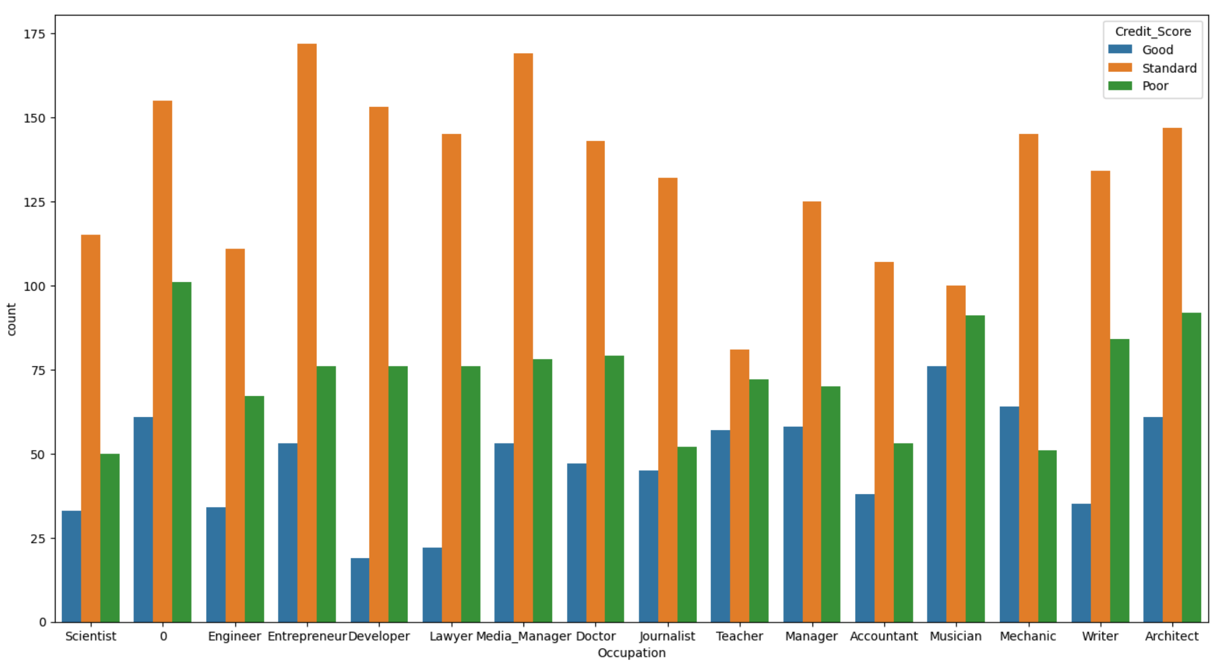


In [499…

```
#Occupaton by payment of min amount
fig = plt.figure(figsize= (17,9))
sns.countplot(data=CRSclean_df,x="Occupation",hue="Credit_Score")
```

Out[499…

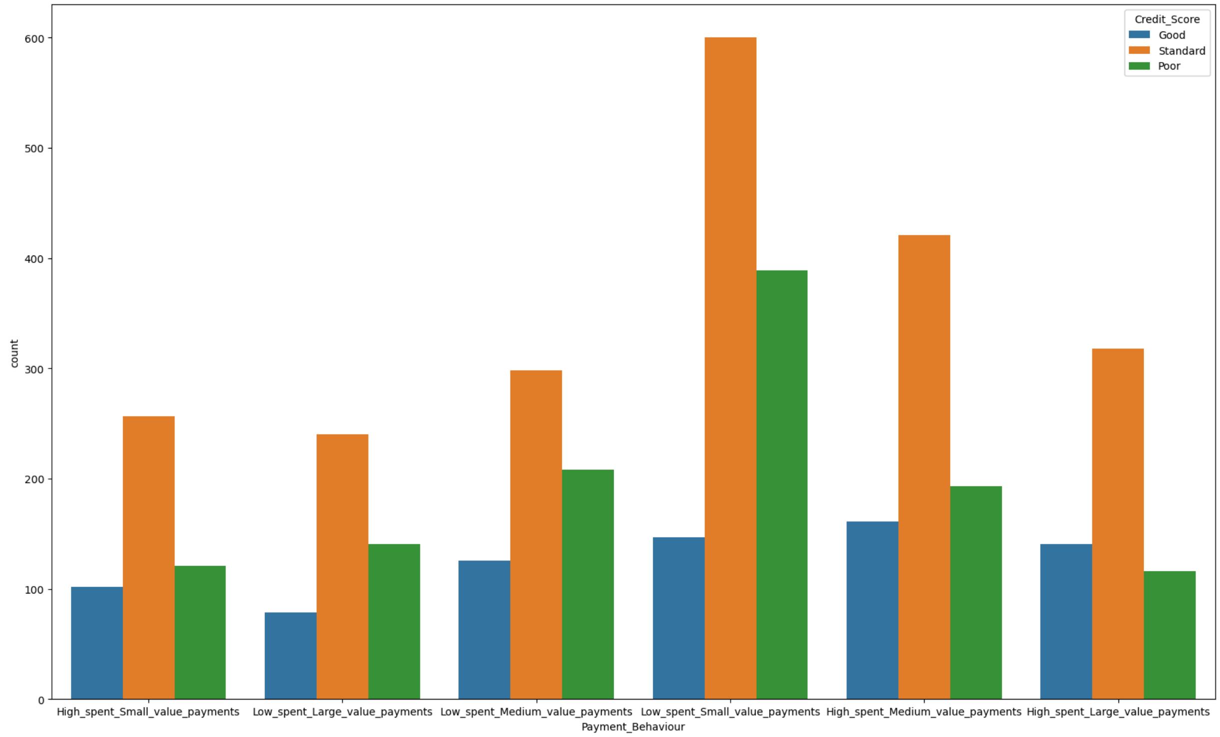
```
<Axes: xlabel='Occupation', ylabel='count'>
```



In [500…

```
#Occupaton by payment of min amount
fig = plt.figure(figsize= (20,12))
sns.countplot(data=CRSclean_df,x="Payment_Behaviour",hue="Credit_Score")
```

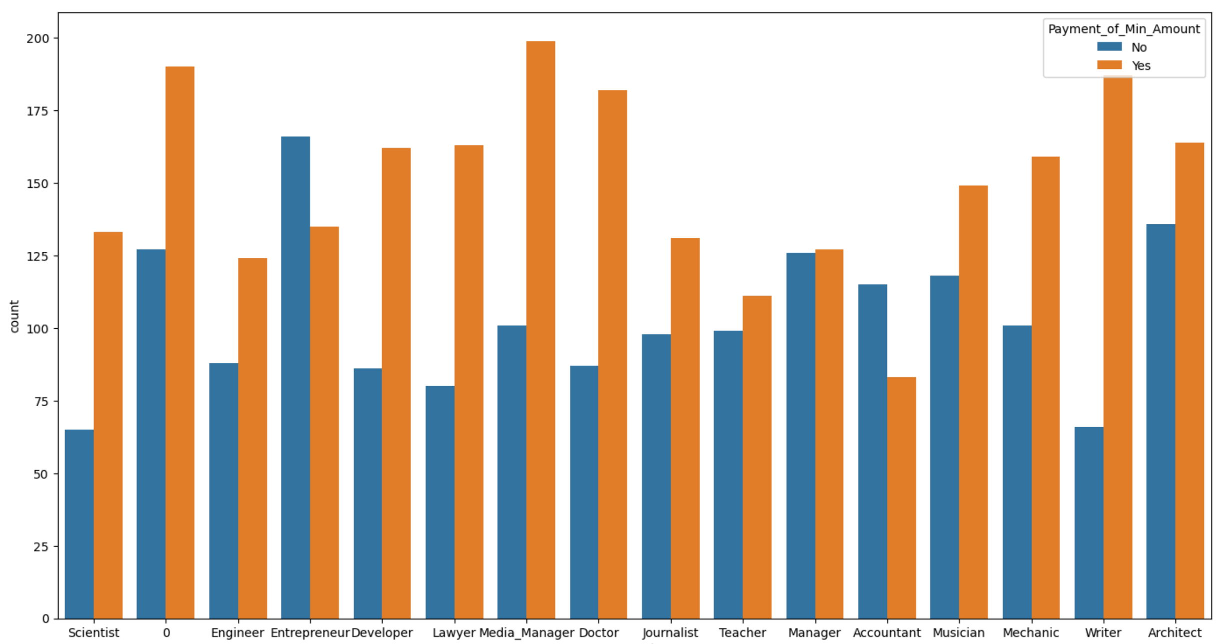
Out[500… <Axes: xlabel='Payment_Behaviour', ylabel='count'>



In [501…

```
#Occupaton by payment of min amount
fig = plt.figure(figsize= (17,9))
sns.countplot(data=CRSclean_df,x="Occupation",hue="Payment_of_Min_Amount")
```

Out[501… <Axes: xlabel='Occupation', ylabel='count'>



In [502…

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
Scaled_data = scaler.fit_transform(CRS_o_df)
```

In [503…

```
column_names = ['Age', 'AnnualIncome', 'Monthly_Inhand_Salary', 'Num_Bank_Accounts'
       'Num_Credit_Card', 'Interest_Rate', 'NumofLoan', 'Delay_from_due_date',
       'Num_of_Delayed_Payment', 'ChangedCreditLimit', 'Num_Credit_Inquiries',
       'OutstandingDebt', 'Credit_Utilization_Ratio', 'Total_EMI_per_month',
       'Amount_invested_monthly', 'Monthly_Balance']
scaled_df = pd.DataFrame(Scaled_data, columns=column_names)
scaled_df = scaled_df.round(4)
print(scaled_df)
```

```
         Age  AnnualIncome  Monthly_Inhand_Salary  Num_Bank_Accounts  \
0    -0.8968       -0.1331                -0.5350            -0.8954   
1    -0.8968       -0.1331                -0.5350            -0.8954   
2    -0.8968       -0.1331                -0.5350            -0.8954   
3    -0.8968       -0.1331                -0.5350            -0.8954   
4    -0.8968       -0.1331                -0.5350            -0.8954   
...      ...           ...                    ...                ...   
4053 -1.1670       -0.0978                 0.7869             0.3677   
4054 -1.1670       -0.0978                 0.7869             0.3677   
4055 -1.1670       -0.0978                 0.7869             0.3677   
4056 -1.1670       -0.0978                 0.7869             0.3677   
4057 -1.1670       -0.0978                 0.7869             0.3677   

      Num_Credit_Card  Interest_Rate  NumofLoan  Delay_from_due_date  \
0             -0.6790        -1.2885     0.2542              -1.2635   
1             -0.6790        -1.2885     0.2542              -1.5357   
2             -0.6790        -1.2885     0.2542              -1.2635   
3             -0.6790        -1.2885     0.2542              -1.1273   
4             -0.6790        -1.2885     0.2542              -1.0593   
...               ...            ...        ...                  ...   
4053           0.4528         0.9978     1.4396               0.7105   
4054           0.4528         0.9978     1.4396               0.5063   
4055           0.4528         0.9978     1.4396               0.6424   
4056           0.4528         0.9978     1.4396               0.6424   
4057           0.4528         0.9978     1.4396               0.6424   

      Num_of_Delayed_Payment  ChangedCreditLimit  Num_Credit_Inquiries  \
0                    -0.1036              0.1219               -0.1091   
1                    -0.0643              0.1219               -0.1091   
2                    -0.1036             -0.1310               -0.1091   
3                    -0.1204             -0.6091               -0.1091   
4                    -0.0643              0.1219               -0.1091   
...                      ...                 ...                   ...   
4053                 -0.0531              0.8968               -0.0960   
4054                 -0.0643              0.8968               -0.0960   
4055                 -0.0475              0.8968               -0.0960   
4056                 -0.0531              0.8968               -0.0960   
4057                 -0.0643              0.8968               -0.0960   

      OutstandingDebt  Credit_Utilization_Ratio  Total_EMI_per_month  \
0             -0.5171                   -1.0600              -0.1600   
1             -0.5171                   -0.0692              -0.1600   
2             -0.5171                   -0.7144              -0.1600   
3             -0.5171                   -0.1789              -0.1600   
4             -0.5171                   -1.4517              -0.1600   
...               ...                       ...                  ...   
4053           1.6053                    0.5537              -0.1199   
4054           1.6053                   -1.8423              -0.1199   
4055           1.6053                   -1.7539              -0.1199   
4056           1.6053                   -1.1542              -0.1199   
4057           1.6053                    1.0347              -0.1199   

      Amount_invested_monthly  Monthly_Balance  
0                     -0.2837          -0.4311  
1                     -0.2652          -0.5624  
2                     -0.2831          -0.3429  
```

```
3                       -0.2254          -0.8507
4                       -0.3028          -0.2944
...                         ...              ...
4053                    -0.1824          -0.7224
4054                    -0.0213          -0.2973
4055                    -0.2895           0.1192
4056                    -0.2509          -0.2520
4057                    -0.0777          -1.7293

[4058 rows x 16 columns]
```

In [504…

```
cat_df = CRSclean_df[cat_cols].copy()
print(cat_df)
```

```
          ID Customer_ID Occupation Payment_of_Min_Amount  \
0     0x1602   CUS_0xd40  Scientist                    No
1     0x1603   CUS_0xd40  Scientist                    No
2     0x1604   CUS_0xd40  Scientist                    No
3     0x1605   CUS_0xd40  Scientist                    No
4     0x1606   CUS_0xd40  Scientist                    No
...      ...         ...        ...                   ...
4992  0x3342  CUS_0x69ea    Manager                   Yes
4994  0x3344  CUS_0x69ea    Manager                   Yes
4996  0x3346  CUS_0x69ea    Manager                   Yes
4998  0x3348  CUS_0x69ea    Manager                   Yes
4999  0x3349  CUS_0x69ea    Manager                   Yes

                   Payment_Behaviour Credit_Score
0      High_spent_Small_value_payments         Good
1       Low_spent_Large_value_payments         Good
2      Low_spent_Medium_value_payments         Good
3       Low_spent_Small_value_payments         Good
4     High_spent_Medium_value_payments         Good
...                                ...          ...
4992   Low_spent_Medium_value_payments     Standard
4994    Low_spent_Small_value_payments     Standard
4996   High_spent_Large_value_payments         Good
4998   High_spent_Large_value_payments         Good
4999   Low_spent_Medium_value_payments         Good

[4058 rows x 6 columns]
```

In [505…

```
scaled_df.reset_index(drop=True, inplace=True)
cat_df.reset_index(drop=True, inplace=True)
combined_df = pd.concat([scaled_df, cat_df], axis=1)
combined_df = combined_df.round(4)
print(combined_df)
```

```
         Age  AnnualIncome  Monthly_Inhand_Salary  Num_Bank_Accounts  \
0    -0.8968       -0.1331                -0.5350            -0.8954   
1    -0.8968       -0.1331                -0.5350            -0.8954   
2    -0.8968       -0.1331                -0.5350            -0.8954   
3    -0.8968       -0.1331                -0.5350            -0.8954   
4    -0.8968       -0.1331                -0.5350            -0.8954   
...      ...           ...                    ...                ...   
4053 -1.1670       -0.0978                 0.7869             0.3677   
4054 -1.1670       -0.0978                 0.7869             0.3677   
4055 -1.1670       -0.0978                 0.7869             0.3677   
4056 -1.1670       -0.0978                 0.7869             0.3677   
4057 -1.1670       -0.0978                 0.7869             0.3677   

      Num_Credit_Card  Interest_Rate  NumofLoan  Delay_from_due_date  \
0             -0.6790        -1.2885     0.2542              -1.2635   
1             -0.6790        -1.2885     0.2542              -1.5357   
2             -0.6790        -1.2885     0.2542              -1.2635   
3             -0.6790        -1.2885     0.2542              -1.1273   
4             -0.6790        -1.2885     0.2542              -1.0593   
...               ...            ...        ...                  ...   
4053           0.4528         0.9978     1.4396               0.7105   
4054           0.4528         0.9978     1.4396               0.5063   
4055           0.4528         0.9978     1.4396               0.6424   
4056           0.4528         0.9978     1.4396               0.6424   
4057           0.4528         0.9978     1.4396               0.6424   

      Num_of_Delayed_Payment  ChangedCreditLimit  ...  \
0                    -0.1036              0.1219  ...   
1                    -0.0643              0.1219  ...   
2                    -0.1036             -0.1310  ...   
3                    -0.1204             -0.6091  ...   
4                    -0.0643              0.1219  ...   
...                      ...                 ...  ...   
4053                 -0.0531              0.8968  ...   
4054                 -0.0643              0.8968  ...   
4055                 -0.0475              0.8968  ...   
4056                 -0.0531              0.8968  ...   
4057                 -0.0643              0.8968  ...   

      Credit_Utilization_Ratio  Total_EMI_per_month  Amount_invested_monthly  \
0                      -1.0600              -0.1600                  -0.2837   
1                      -0.0692              -0.1600                  -0.2652   
2                      -0.7144              -0.1600                  -0.2831   
3                      -0.1789              -0.1600                  -0.2254   
4                      -1.4517              -0.1600                  -0.3028   
...                        ...                  ...                      ...   
4053                    0.5537              -0.1199                  -0.1824   
4054                   -1.8423              -0.1199                  -0.0213   
4055                   -1.7539              -0.1199                  -0.2895   
4056                   -1.1542              -0.1199                  -0.2509   
4057                    1.0347              -0.1199                  -0.0777   

      Monthly_Balance      ID  Customer_ID Occupation Payment_of_Min_Amount  \
0             -0.4311  0x1602    CUS_0xd40  Scientist                    No   
1             -0.5624  0x1603    CUS_0xd40  Scientist                    No   
2             -0.3429  0x1604    CUS_0xd40  Scientist                    No   
```

```
3              -0.8507  0x1605    CUS_0xd40  Scientist                        No
4              -0.2944  0x1606    CUS_0xd40  Scientist                        No
...                ...      ...         ...        ...                       ...
4053           -0.7224  0x3342   CUS_0x69ea    Manager                       Yes
4054           -0.2973  0x3344   CUS_0x69ea    Manager                       Yes
4055            0.1192  0x3346   CUS_0x69ea    Manager                       Yes
4056           -0.2520  0x3348   CUS_0x69ea    Manager                       Yes
4057           -1.7293  0x3349   CUS_0x69ea    Manager                       Yes

                     Payment_Behaviour Credit_Score
0      High_spent_Small_value_payments         Good
1       Low_spent_Large_value_payments         Good
2      Low_spent_Medium_value_payments         Good
3       Low_spent_Small_value_payments         Good
4     High_spent_Medium_value_payments         Good
...                                ...          ...
4053   Low_spent_Medium_value_payments     Standard
4054    Low_spent_Small_value_payments     Standard
4055   High_spent_Large_value_payments         Good
4056   High_spent_Large_value_payments         Good
4057   Low_spent_Medium_value_payments         Good

[4058 rows x 22 columns]
```

In [506…

```
combined_df['Credit_Score'].replace({"Poor":0, "Standard":1, "Good":2}, inplace=Tru
combined_df['Payment_of_Min_Amount'].replace({"Yes":1, "No":0}, inplace=True)
combined_df = pd.get_dummies(combined_df, columns = ['Occupation', 'Payment_Behavio
print(combined_df)
```

```
         Age  AnnualIncome  Monthly_Inhand_Salary  Num_Bank_Accounts  \
0    -0.8968       -0.1331                -0.5350            -0.8954   
1    -0.8968       -0.1331                -0.5350            -0.8954   
2    -0.8968       -0.1331                -0.5350            -0.8954   
3    -0.8968       -0.1331                -0.5350            -0.8954   
4    -0.8968       -0.1331                -0.5350            -0.8954   
...      ...           ...                    ...                ...   
4053 -1.1670       -0.0978                 0.7869             0.3677   
4054 -1.1670       -0.0978                 0.7869             0.3677   
4055 -1.1670       -0.0978                 0.7869             0.3677   
4056 -1.1670       -0.0978                 0.7869             0.3677   
4057 -1.1670       -0.0978                 0.7869             0.3677   

      Num_Credit_Card  Interest_Rate  NumofLoan  Delay_from_due_date  \
0             -0.6790        -1.2885     0.2542              -1.2635   
1             -0.6790        -1.2885     0.2542              -1.5357   
2             -0.6790        -1.2885     0.2542              -1.2635   
3             -0.6790        -1.2885     0.2542              -1.1273   
4             -0.6790        -1.2885     0.2542              -1.0593   
...               ...            ...        ...                  ...   
4053           0.4528         0.9978     1.4396               0.7105   
4054           0.4528         0.9978     1.4396               0.5063   
4055           0.4528         0.9978     1.4396               0.6424   
4056           0.4528         0.9978     1.4396               0.6424   
4057           0.4528         0.9978     1.4396               0.6424   

      Num_of_Delayed_Payment  ChangedCreditLimit  ...  Occupation_Musician  \
0                    -0.1036              0.1219  ...                False   
1                    -0.0643              0.1219  ...                False   
2                    -0.1036             -0.1310  ...                False   
3                    -0.1204             -0.6091  ...                False   
4                    -0.0643              0.1219  ...                False   
...                      ...                 ...  ...                  ...   
4053                 -0.0531              0.8968  ...                False   
4054                 -0.0643              0.8968  ...                False   
4055                 -0.0475              0.8968  ...                False   
4056                 -0.0531              0.8968  ...                False   
4057                 -0.0643              0.8968  ...                False   

      Occupation_Scientist  Occupation_Teacher  Occupation_Writer  \
0                     True               False              False   
1                     True               False              False   
2                     True               False              False   
3                     True               False              False   
4                     True               False              False   
...                    ...                 ...                ...   
4053                 False               False              False   
4054                 False               False              False   
4055                 False               False              False   
4056                 False               False              False   
4057                 False               False              False   

      Payment_Behaviour_High_spent_Large_value_payments  \
0                                                 False   
1                                                 False   
2                                                 False   
```

```
3                                                   False
4                                                   False
...                                                   ...
4053                                                False
4054                                                False
4055                                                 True
4056                                                 True
4057                                                False

      Payment_Behaviour_High_spent_Medium_value_payments  \
0                                                   False
1                                                   False
2                                                   False
3                                                   False
4                                                    True
...                                                   ...
4053                                                False
4054                                                False
4055                                                False
4056                                                False
4057                                                False

     Payment_Behaviour_High_spent_Small_value_payments  \
0                                                  True
1                                                 False
2                                                 False
3                                                 False
4                                                 False
...                                                 ...
4053                                              False
4054                                              False
4055                                              False
4056                                              False
4057                                              False

     Payment_Behaviour_Low_spent_Large_value_payments  \
0                                                False
1                                                 True
2                                                False
3                                                False
4                                                False
...                                                ...
4053                                             False
4054                                             False
4055                                             False
4056                                             False
4057                                             False

      Payment_Behaviour_Low_spent_Medium_value_payments  \
0                                                   False
1                                                   False
2                                                    True
3                                                   False
4                                                   False
...                                                   ...
4053                                                 True
```

```
4054                                                  False
4055                                                  False
4056                                                  False
4057                                                   True

      Payment_Behaviour_Low_spent_Small_value_payments
0                                                    False
1                                                    False
2                                                    False
3                                                     True
4                                                    False
...                                                    ...
4053                                                 False
4054                                                  True
4055                                                 False
4056                                                 False
4057                                                 False

[4058 rows x 42 columns]
```

In [507…

```
#Feature selection
```

In [508…

```
X = combined_df.drop(['Credit_Score','ID','Customer_ID'] , axis=1)  # Features
y = combined_df['Credit_Score']
```

In [509…

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
```

In [510…

```
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
```

In [511…

```
rf_classifier.fit(X_train, y_train)
```

Out[511…

```
▾ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [512…

```
#  want to see the importance of each feature
feature_importances = pd.Series(rf_classifier.feature_importances_, index=X.columns
# Round the feature importances to 2 decimal places and sort them in descending ord
rounded_feature_importances = feature_importances.round(4).sort_values(ascending=Fa
# Print the rounded and sorted feature importances
print(rounded_feature_importances)
```

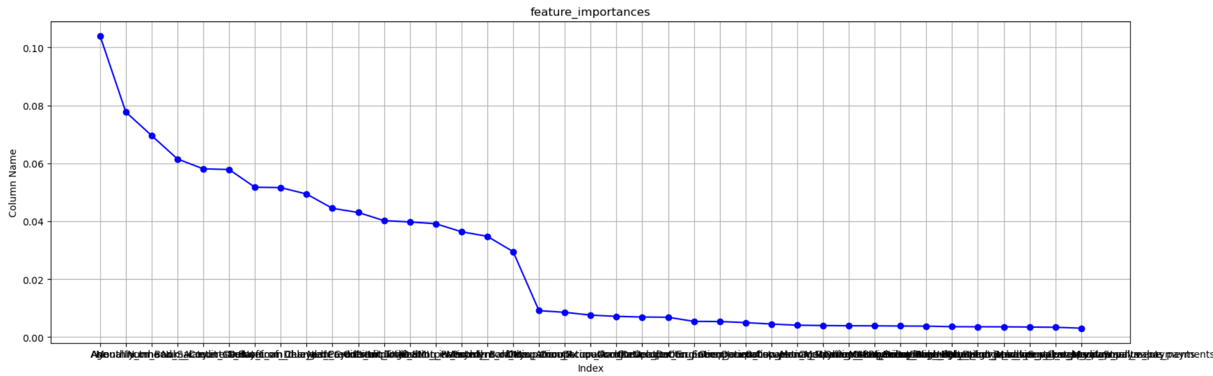
```
OutstandingDebt                                      0.1041
Interest_Rate                                        0.0777
Delay_from_due_date                                  0.0696
Num_of_Delayed_Payment                               0.0615
ChangedCreditLimit                                   0.0581
Monthly_Balance                                      0.0579
Credit_Utilization_Ratio                             0.0517
Amount_invested_monthly                              0.0516
Num_Credit_Inquiries                                 0.0494
AnnualIncome                                         0.0445
Total_EMI_per_month                                  0.0430
Monthly_Inhand_Salary                                0.0402
Num_Credit_Card                                      0.0398
Age                                                  0.0392
Payment_of_Min_Amount                                0.0364
Num_Bank_Accounts                                    0.0348
NumofLoan                                            0.0295
Payment_Behaviour_High_spent_Medium_value_payments   0.0092
Payment_Behaviour_Low_spent_Small_value_payments     0.0086
Payment_Behaviour_High_spent_Large_value_payments    0.0076
Payment_Behaviour_Low_spent_Medium_value_payments    0.0072
Payment_Behaviour_Low_spent_Large_value_payments     0.0069
Payment_Behaviour_High_spent_Small_value_payments    0.0069
Occupation_Musician                                  0.0054
Occupation_0                                         0.0054
Occupation_Entrepreneur                              0.0050
Occupation_Accountant                                0.0045
Occupation_Mechanic                                  0.0041
Occupation_Lawyer                                    0.0040
Occupation_Manager                                   0.0039
Occupation_Doctor                                    0.0039
Occupation_Developer                                 0.0038
Occupation_Engineer                                  0.0038
Occupation_Scientist                                 0.0036
Occupation_Writer                                    0.0036
Occupation_Architect                                 0.0036
Occupation_Journalist                                0.0035
Occupation_Media_Manager                             0.0034
Occupation_Teacher                                   0.0031
dtype: float64
```

In [513…

```
#Going forward with these feature : OutstandingDebt, Interest_Rate ,Delay_from_due_
#Num_of_Delayed_Payment, ChangedCreditLimit, Monthly_Balance, Credit_Utilization_Ra
```

In [514…

```
#Graph for feature selection
plt.figure(figsize=(20, 6))
plt.plot(feature_importances.index, feature_importances.sort_values(ascending=False
plt.title('feature_importances')
plt.xlabel('Index')
plt.ylabel('Column Name')
plt.grid(True)
plt.show()
```



In [515...
```
#Specifying x and Y
X1 = combined_df[['OutstandingDebt', 'Interest_Rate' ,'Delay_from_due_date', 'Num_o
y1 = combined_df['Credit_Score']
```

In [516...
```
X1_train,X1_test,y1_train,y1_test= train_test_split(X1, y1, test_size=0.2, random_s
```

In [517...
```
#MODEL ANALYSIS
```

1. Fitting KNN Model

In [518...
```
#Fitting the KNN model with three nearset neigbhours
knn=KNeighborsClassifier(n_neighbors=3)
knn.fit(X1_train,np.ravel(y1_train))
```

Out[518...
```
▾ KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

In [519...
```
# Predicting on validation data
knn_pred=knn.predict(X1_test)
```

In [520...
```
#checking the accuracy score
accuracy=accuracy_score(y1_test,knn_pred)
print(f'accuracy:{accuracy}')
```

```
accuracy:0.604679802955665
```

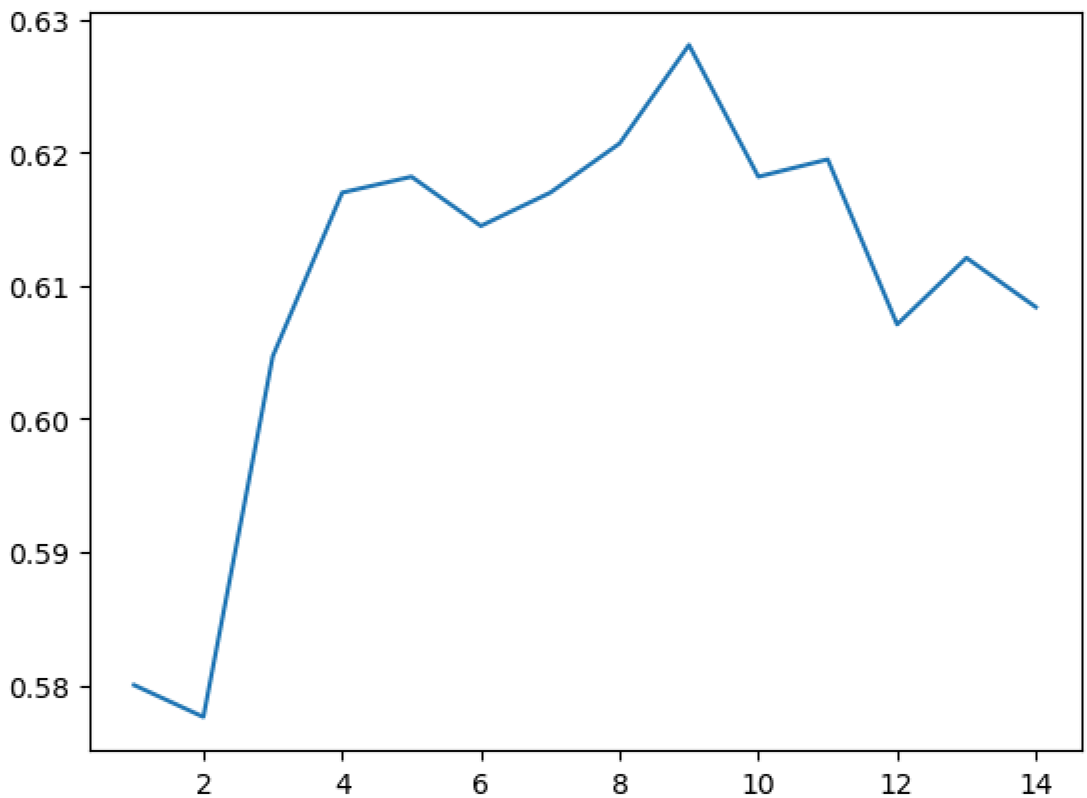
In [521...
```
# Train a classifier for different values of k
results = []
for k in range(1, 15):
     knn = KNeighborsClassifier(n_neighbors=k).fit(X1_train, y1_train)
     results.append({
    'k': k,
     'accuracy': accuracy_score(y1_test, knn.predict(X1_test))
})
```

In [522...
```
# Convert results to a pandas data frame
results = pd.DataFrame(results).round(4)
print(results)
```

```
     k  accuracy
0    1    0.5800
1    2    0.5776
2    3    0.6047
3    4    0.6170
4    5    0.6182
5    6    0.6145
6    7    0.6170
7    8    0.6207
8    9    0.6281
9   10    0.6182
10  11    0.6195
11  12    0.6071
12  13    0.6121
13  14    0.6084
```

In [523…
```
plt.plot(results["k"],results["accuracy"])
```

Out[523…
```
[<matplotlib.lines.Line2D at 0x19cbf14f510>]
```



K=9 is the optimal choice of nearest neighbours since its provides highest accuracy i.e. 63%

In [524…
```
# Tuning the Model by taking K=9
knn = KNeighborsClassifier(n_neighbors=9)
knn.fit(X1_train,y1_train)
knn_pred=knn.predict(X1_test)
accuracy_1=accuracy_score(y1_test,knn_pred)
print("Accuracy of the KNN Model where k=9 is: ",round(results["accuracy"].max()*10
```

```
Accuracy of the KNN Model where k=9 is:  62.81 %
```

2. Decision Tree

In [525…

```
# fitthing Decision tree
Tree=DecisionTreeClassifier(max_depth=2)
Tree.fit(X1_train,y1_train)
```

Out[525…

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(max_depth=2)
```

In [526…

```
# making prediction on test data
Tree_pred=Tree.predict(X1_test)
print(Tree_pred)
```

```
[2 1 1 2 1 1 1 1 1 2 2 1 1 1 1 2 1 0 0 1 2 1 0 0 1 1 2 0 1 1 1 1 1 1 1 0 0
 2 1 1 2 0 1 1 1 0 1 1 1 1 1 0 1 0 0 1 1 0 2 1 2 1 1 1 1 1 1 1 2 1 1 1 2 1
 2 1 2 1 1 1 1 1 0 2 2 1 1 1 2 1 2 1 1 1 1 2 0 1 2 1 1 1 2 0 1 1 1 1 2 1 1
 1 1 2 1 1 1 0 2 1 0 1 1 1 1 1 2 1 2 0 2 0 0 0 1 1 1 1 0 2 1 1 1 1 1 1 1 0
 2 0 0 1 2 1 1 1 1 1 0 1 0 1 1 1 2 1 0 1 2 0 2 1 2 1 2 2 0 0 1 0 1 1 1 1 2
 0 1 1 1 1 1 2 1 2 2 1 1 0 1 1 1 1 1 1 2 1 2 1 1 1 0 1 1 2 2 1 2 2 1 1 0 2
 1 1 0 2 0 1 2 0 0 1 1 0 1 1 2 1 1 2 1 1 1 2 0 2 2 2 0 1 1 2 0 0 2 1 1 1 1
 1 1 0 2 2 1 2 0 1 1 2 1 1 2 1 2 1 1 1 1 1 0 2 1 2 1 2 1 1 1 1 1 1 1 0 1 0
 1 1 2 2 0 1 0 1 0 1 1 2 1 1 2 1 2 1 2 1 0 2 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1
 1 1 1 0 2 1 0 1 1 0 0 1 1 1 1 1 2 1 1 1 2 2 1 0 1 1 1 1 1 0 1 1 1 1 1 2 1
 1 0 1 2 1 0 1 1 2 0 2 1 1 1 1 1 1 2 0 1 1 1 2 1 1 1 1 2 0 0 0 1 1 1 1 1 0
 1 1 1 2 2 1 1 1 2 1 1 2 1 1 1 0 1 1 1 1 2 1 2 1 2 1 1 1 1 0 2 0 1 0 1 0 1
 1 2 2 1 1 1 1 2 1 0 1 1 0 1 1 1 0 2 0 1 2 1 1 2 2 2 1 1 0 2 1 1 2 2 2 1 1
 0 2 2 1 1 1 1 1 1 1 1 2 2 2 1 1 1 1 0 2 1 1 1 1 1 1 1 0 1 1 2 1 1 2 1 2 1
 0 1 0 1 1 2 1 1 1 1 0 0 1 0 1 1 2 0 1 1 2 1 2 2 0 2 2 1 1 0 2 1 2 1 1 1 1
 0 1 0 1 0 1 1 0 0 2 1 1 1 0 2 0 1 0 1 0 0 2 2 0 1 1 2 2 0 2 0 1 2 1 1 1 2
 1 1 1 2 1 1 1 1 0 0 0 1 1 2 1 1 1 0 1 1 1 2 2 1 1 0 1 1 1 1 1 1 1 1 2 1 1
 1 2 1 1 1 0 1 2 1 1 1 0 0 1 0 1 0 1 1 1 2 1 1 2 1 2 1 1 2 1 1 1 1 1 1 1 0
 1 1 1 1 1 2 2 1 2 1 1 2 1 2 1 1 2 0 1 1 1 1 1 0 2 1 1 0 1 1 2 1 1 1 1 1 1
 1 0 2 2 2 2 1 2 0 2 1 0 1 1 1 1 0 2 2 1 2 2 2 1 1 1 1 1 2 1 1 0 0 1 2 1 1
 1 0 2 1 2 2 1 1 2 1 2 0 1 1 1 2 1 1 1 1 0 1 1 1 1 1 1 1 2 1 0 0 0 1 1 1 1
 2 0 1 1 0 1 1 1 2 1 1 2 1 0 1 2 1 1 1 1 1 2 1 1 1 1 2 1 2 1 1 1 1 1 0]
```

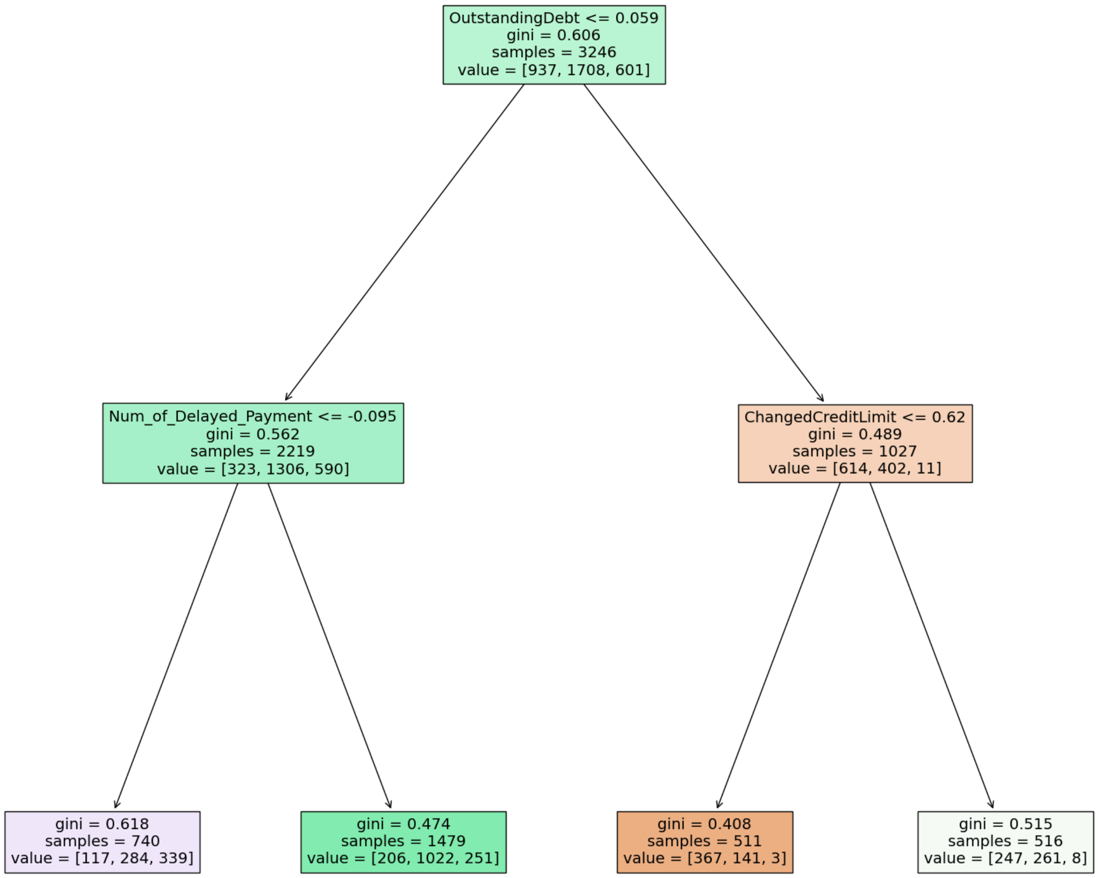
In [527…

```
accuracy_2=accuracy_score(y1_test,Tree_pred)
print(accuracy_2)
```

```
0.604679802955665
```

In [528…

```
# Graph
fig=plt.figure(figsize=(20,20))
_=tree.plot_tree(Tree,feature_names=X1.columns,filled=True)
```



3. Random Forest

In [529…

```
# Initialize the Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier
clf.fit(X1_train, y1_train)
```

Out[529…

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [530…

```
# Make predictions on the test set
y_pred = clf.predict(X1_test)
```

In [531…

```
# Evaluate the model
accuracy_3 = accuracy_score(y1_test, y_pred)
```

```
print(f'Accuracy: {accuracy_3:.2f}')
```

```
Accuracy: 0.75
```

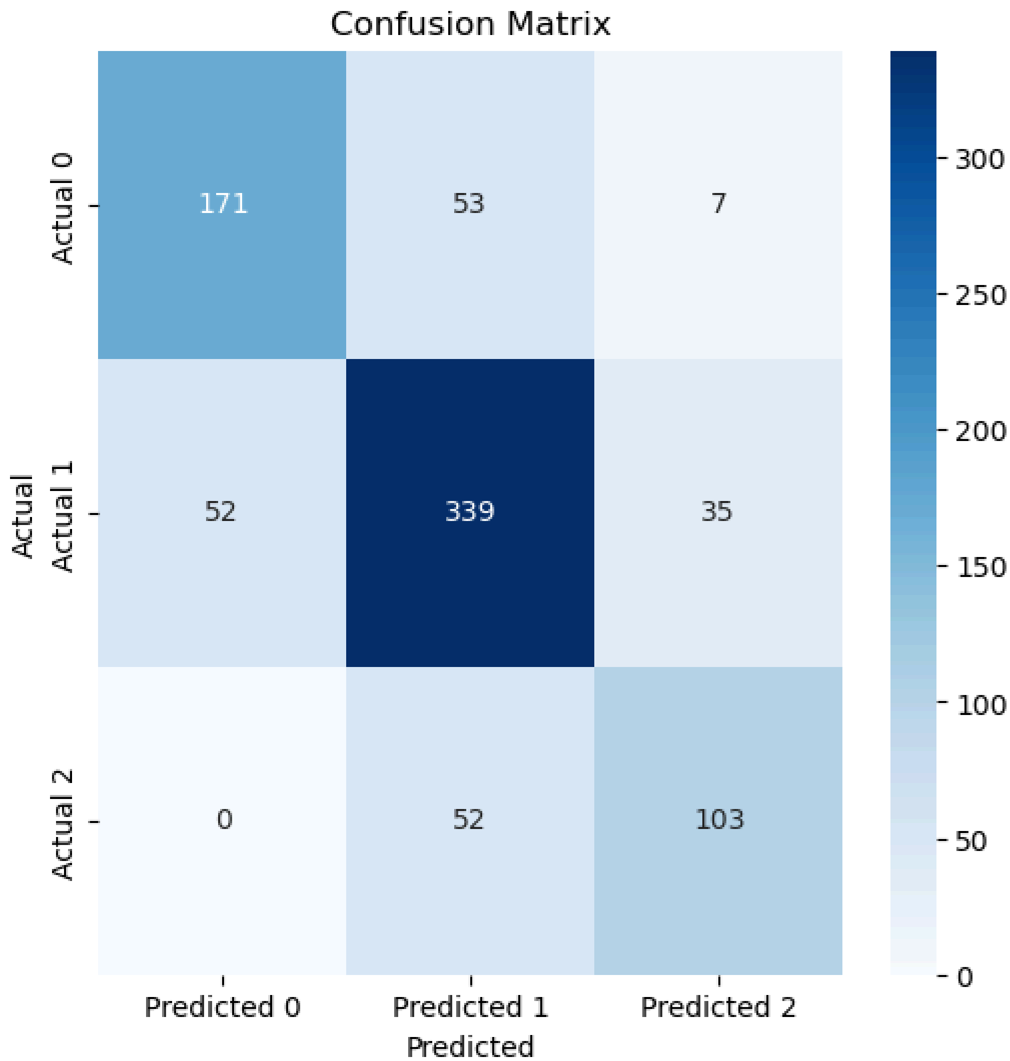
In [532…

```
# Print detailed classification report
print(classification_report(y1_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.77      0.74      0.75       231
           1       0.76      0.80      0.78       426
           2       0.71      0.66      0.69       155

    accuracy                           0.75       812
   macro avg       0.75      0.73      0.74       812
weighted avg       0.75      0.75      0.75       812
```

In [533…

```
# Compute and plot confusion matrix
cm = confusion_matrix(y1_test, y_pred)
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Predicted 0', 'Pre
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



4. Hierarchical Clustering

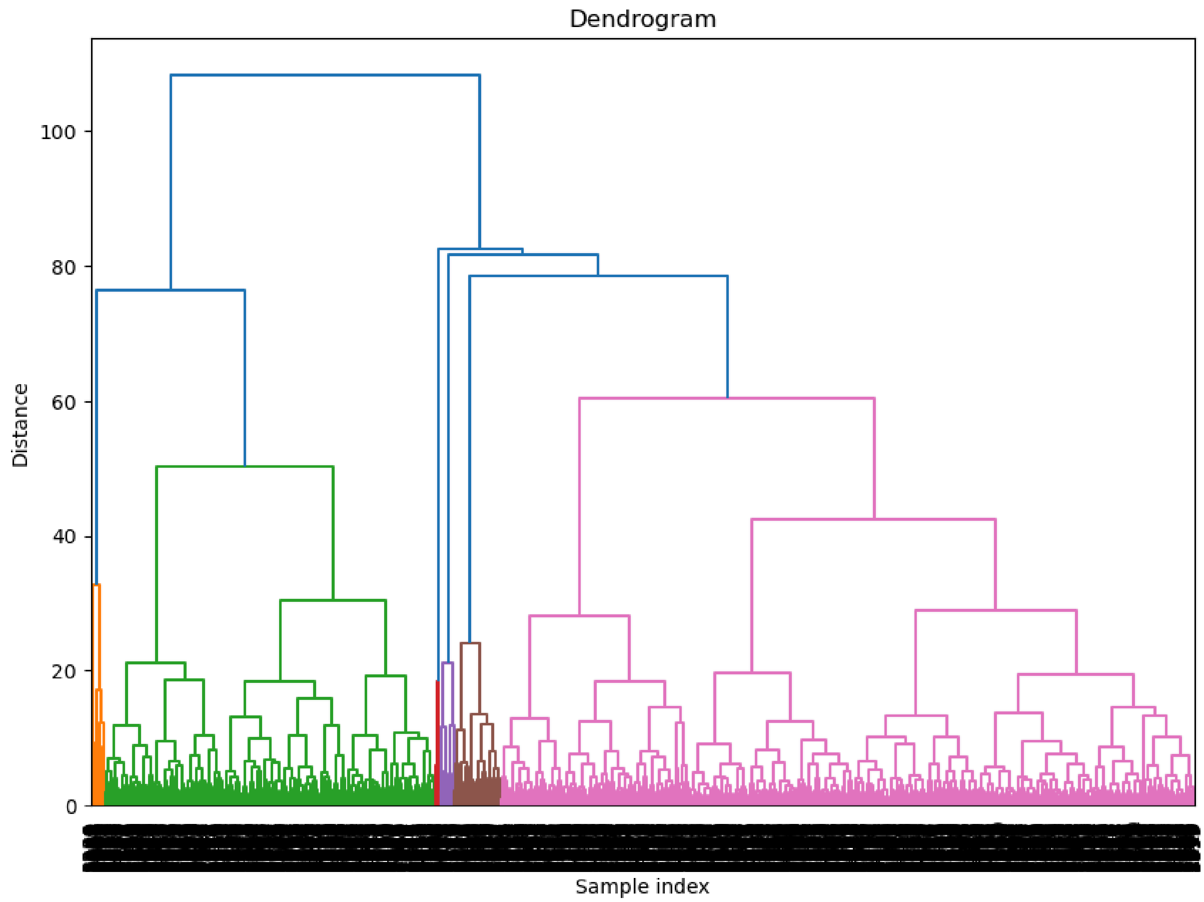
In [534…

```
# perform Hierarchical Clustering
Z = linkage(X1_train, method='ward')  # Other methods: 'single', 'complete', 'avera
```

accuracy = accuracy_score(y1_train, clusters) print(f'Clustering accuracy: {accuracy:.2f}')

In [535…

```
# Plot the dendrogram
plt.figure(figsize=(10, 7))
dendrogram(Z, labels=X1_train.index, leaf_rotation=90, leaf_font_size=10)
plt.title('Dendrogram')
plt.xlabel('Sample index')
plt.ylabel('Distance')
plt.show()
```



In [536…

```
# Cut the dendrogram to form flat clusters
# The threshold can be set to determine the number of clusters
# Here, 't' is the threshold and 'criterion' specifies how the threshold is applied
clusters = fcluster(Z, t=3, criterion='maxclust')  # Form 3 clusters
```

In [537…

```
# Evaluate the Clustering
ari = adjusted_rand_score(y1_train, clusters)
nmi = normalized_mutual_info_score(y1_train, clusters)

print(f'Adjusted Rand Index: {ari:.2f}')
print(f'Normalized Mutual Information: {nmi:.2f}')
```

```
Adjusted Rand Index: 0.13
Normalized Mutual Information: 0.15
```

Interpretation: An ARI of 0.13 indicates a slight positive correlation between the clustering results and the true labels, but the clustering is not very effective. The clusters found by the hierarchical clustering algorithm do not align well with the actual classes.

An NMI of 0.15 indicates that there is some mutual information between the clustering results and the true labels, but it is quite low. This means that the clusters share some information with the actual classes, but overall, the clustering does not capture the true class structure well.

In [538…

```
# Add the cluster labels to the original training data
df_train = X_train.copy()
df_train['Cluster'] = clusters
print(df_train)
```

```
         Age  AnnualIncome  Monthly_Inhand_Salary  Num_Bank_Accounts  \
2279  0.4543       -0.0945                 1.0906             1.2097   
3570  0.1841       -0.1327                -0.6699             0.3677   
436   0.8146       -0.1092                -1.0883             1.2097   
3486  0.9046       -0.1322                -0.6230            -2.1584   
3652  1.0848       -0.1080                 0.3806            -0.8954   
...      ...           ...                    ...                ...   
1130 -0.7167       -0.1276                -1.0883             1.2097   
1294 -0.5365       -0.1206                -0.1147            -0.0533   
860   0.4543       -0.1202                -0.0702             0.3677   
3507  0.8146       -0.0809                 1.5177            -0.4744   
3174 -0.7167       -0.0762                 1.7379            -1.7374   

      Num_Credit_Card  Interest_Rate  NumofLoan  Delay_from_due_date  \
2279           1.0187         0.8835     1.0445               0.9147   
3570           0.4528         0.8835     1.0445               0.2340   
436           -0.1131         0.0833     0.2542               2.7525   
3486           1.0187        -1.4028    -0.9312              -0.5828   
3652          -1.2449         0.6549    -0.5360               0.5063   
...               ...            ...        ...                  ...   
1130           0.4528         2.0266     1.4396               1.8676   
1294          -0.1131        -0.4883    -0.9312              -0.7189   
860            1.0187         0.4262    -0.1409               0.9147   
3507           0.4528        -1.0598     0.2542              -0.8550   
3174          -0.6790        -0.9455     0.2542              -1.1954   

      Num_of_Delayed_Payment  ChangedCreditLimit  ...  Occupation_Scientist  \
2279                 -0.0195              1.6907  ...                 False   
3570                 -0.0419              0.8983  ...                 False   
436                  -0.0643             -1.0448  ...                 False   
3486                 -0.1260              0.1833  ...                 False   
3652                 -0.0475              0.8822  ...                 False   
...                      ...                 ...  ...                   ...   
1130                 -0.0475              1.7155  ...                 False   
1294                 -0.1260             -1.2363  ...                 False   
860                  -0.0139             -1.0126  ...                 False   
3507                 -0.0531             -0.4921  ...                 False   
3174                 -0.1372             -1.0287  ...                 False   

      Occupation_Teacher  Occupation_Writer  \
2279               False              False   
3570               False              False   
436                False              False   
3486               False              False   
3652               False              False   
...                  ...                ...   
1130               False              False   
1294               False              False   
860                False              False   
3507               False              False   
3174               False              False   

      Payment_Behaviour_High_spent_Large_value_payments  \
2279                                              False   
3570                                              False   
436                                               False   
```

```
3486                                              False
3652                                               True
...                                                 ...
1130                                              False
1294                                              False
860                                               False
3507                                              False
3174                                              False

      Payment_Behaviour_High_spent_Medium_value_payments  \
2279                                              False
3570                                              False
436                                               False
3486                                              False
3652                                              False
...                                                 ...
1130                                              False
1294                                              False
860                                                True
3507                                               True
3174                                              False

      Payment_Behaviour_High_spent_Small_value_payments  \
2279                                              False
3570                                              False
436                                               False
3486                                              False
3652                                              False
...                                                 ...
1130                                              False
1294                                              False
860                                               False
3507                                              False
3174                                              False

      Payment_Behaviour_Low_spent_Large_value_payments  \
2279                                             False
3570                                             False
436                                              False
3486                                             False
3652                                             False
...                                                ...
1130                                             False
1294                                              True
860                                              False
3507                                             False
3174                                              True

      Payment_Behaviour_Low_spent_Medium_value_payments  \
2279                                               True
3570                                               True
436                                                True
3486                                               True
3652                                              False
...                                                 ...
1130                                              False
```

```
1294                                                  False
860                                                   False
3507                                                  False
3174                                                  False

      Payment_Behaviour_Low_spent_Small_value_payments  Cluster
2279                                               False        1
3570                                               False        1
436                                                False        1
3486                                               False        3
3652                                               False        1
...                                                  ...      ...
1130                                                True        3
1294                                               False        3
860                                                False        1
3507                                               False        3
3174                                               False        3

[3246 rows x 40 columns]
```

5. Naive Bayes

In [539…

```
# Assumptions: 1. Assume Independence of predictor Varaiable. 2. Requires Categoric
```

In [540…

```
# converting numerical data to categorical
X2_train=X1_train.astype('category')
y2_train=y1_train.astype('category')

X2_test=X1_test.astype('category')
y2_test=y1_test.astype('category')
```

Since the dataset contains negative values, we need to standardised the dataset by MinMaxScaler to perform Naive Bayes.

In [541…

```
scaler = MinMaxScaler(feature_range=(0, 1))
X2_train_scaled = scaler.fit_transform(X2_train)
X2_test_scaled = scaler.fit_transform(X2_test)
```

In [542…

```
nb=MultinomialNB(alpha=0.01)
nb.fit(X2_train_scaled,y2_train)
```

Out[542…

```
▾ MultinomialNB
MultinomialNB(alpha=0.01)
```

In [543…

```
# predict probabilities (Shows the belonging probabilities of each record to which
predProb_train = nb.predict_proba(X2_train_scaled)
print(predProb_train)
```

```
predProb_test = nb.predict_proba(X2_test_scaled)
print(predProb_test)
```

```
[[0.34401843 0.54037784 0.11560373]
 [0.32670842 0.54427312 0.12901846]
 [0.32522251 0.53347013 0.14130736]
 ...
 [0.32982491 0.53376814 0.13640696]
 [0.22619835 0.53662288 0.23717878]
 [0.25976613 0.53713436 0.20309952]]
[[0.28205275 0.55167569 0.16627157]
 [0.2486806  0.5514735  0.1998459 ]
 [0.34997626 0.54061404 0.1094097 ]
 ...
 [0.28401396 0.56127365 0.15471239]
 [0.27431621 0.54934796 0.17633583]
 [0.34027054 0.54137408 0.11835538]]
```

In [544…

```
# predict class membership (shows the class instead of probability by selecting the
y_test_pred = nb.predict(X2_test_scaled)
print(y_test_pred)
y_train_pred = nb.predict(X2_train_scaled)
print(y_train_pred)
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
[1 1 1 ... 1 1 1]
```

In [545…

```
accuracy_4 = accuracy_score(y2_test, y_test_pred)
print("Accuary is",accuracy_4)


# Confusion Matrix
conf_matrix = confusion_matrix(y2_test, y_test_pred)
print('Confusion Matrix:')
print(conf_matrix)
```

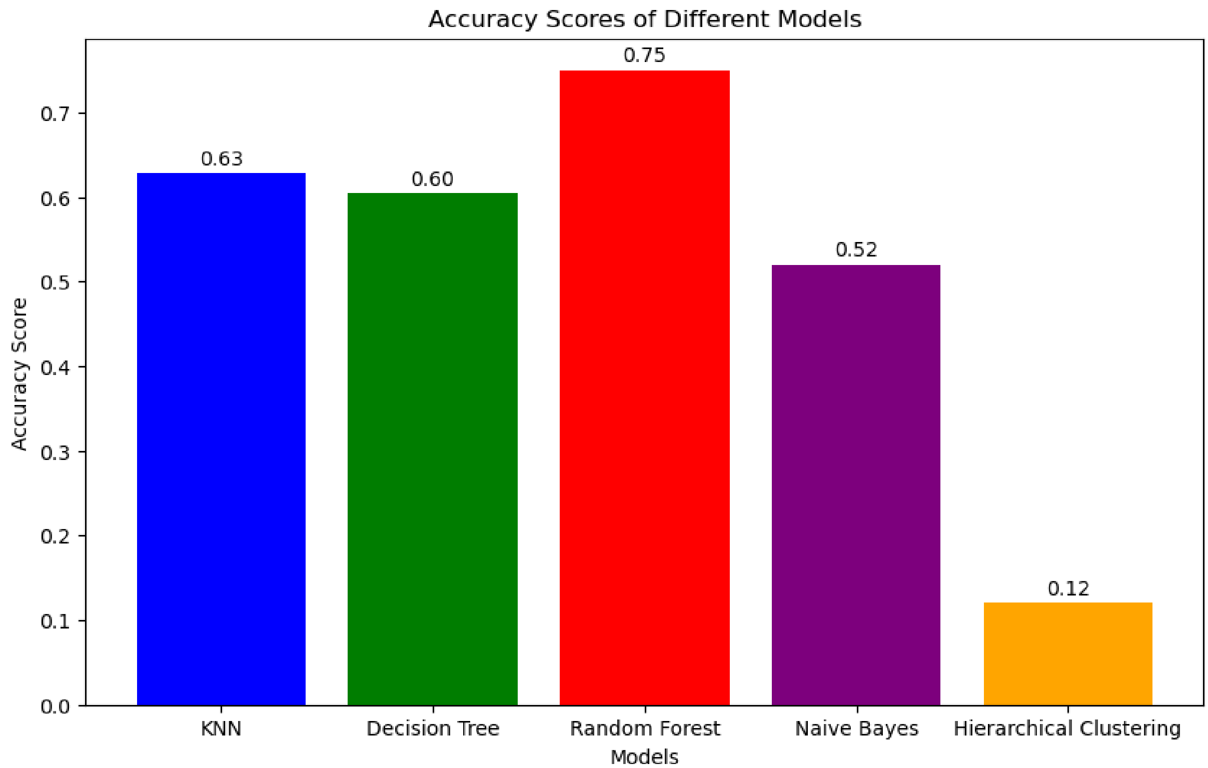
```
# Classification Report
class_report = classification_report(y2_test, y_test_pred)
print('Classification Report:')
print(class_report)
```

```
Accuary is 0.5233990147783252
Confusion Matrix:
[[  0 231   0]
 [  1 425   0]
 [  0 155   0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       231
           1       0.52      1.00      0.69       426
           2       0.00      0.00      0.00       155

    accuracy                           0.52       812
   macro avg       0.17      0.33      0.23       812
weighted avg       0.27      0.52      0.36       812
```

```
E:\ML-anaconda\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedM
etricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels w
ith no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\ML-anaconda\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedM
etricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels w
ith no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
E:\ML-anaconda\Lib\site-packages\sklearn\metrics\_classification.py:1344: UndefinedM
etricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels w
ith no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [546…

```
#Comparison between different models
import matplotlib.pyplot as plt
model_names = ['KNN', 'Decision Tree', 'Random Forest', 'Naive Bayes','Hierarchical
accuracy_scores = [0.6281, 0.6046,  0.75, 0.52,0.12]
plt.figure(figsize=(10, 6))
plt.bar(model_names, accuracy_scores, color=['blue', 'green', 'red', 'purple','oran
plt.title('Accuracy Scores of Different Models')
plt.xlabel('Models')
plt.ylabel('Accuracy Score')

# Display the accuracy scores on top of the bars
for i, score in enumerate(accuracy_scores):
    plt.text(i, score + 0.01, f'{score:.2f}', ha='center')
plt.show()
```


Accuracy Scores of Different Models
Accuracy Score
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.63
0.60
0.75
0.52
0.12
KNN
Decision Tree
Random Forest
Naive Bayes
Hierarchical Clustering
Models

In [ ]: