jupyter Untitled7 Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help

Code JupyterLab Pytho

[13]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import roc_auc_score, roc_curve
```

[ ]:
```
#LOAD DATA
```

[ ]:
```
#PREVIEW DATA
```

[5]:
```
df1 = pd.read_csv(r"C:\Users\LENOVO\Desktop\kaggle practice data\Loan Dataset.csv")
df1.head()
```

[5]:

	Applicant_ID	Gender	Age	Marital_Status	Dependents	Education	Employment_Status	Occupation_Type	Residential_Status	City/Town
0	1	Female	25	Married	2	Graduate	Employed	Business	Own	Urban
1	2	Male	36	Married	2	High School	Employed	Business	Own	Suburban
2	3	Male	43	Single	0	Postgraduate	Self-Employed	Freelancer	Own	Urban
3	4	Female	28	Married	0	High School	Self-Employed	Freelancer	Rent	Suburban
4	5	Female	32	Single	0	Graduate	Employed	Salaried	Rent	Suburban

5 rows × 27 columns

[ ]:
```
#GET DATA INFORMATION
```

[17]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52000 entries, 0 to 51999
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Applicant_ID               52000 non-null  int64
 1   Gender                     52000 non-null  object
 2   Age                        52000 non-null  int64
 3   Marital_Status             52000 non-null  object
 4   Dependents                 52000 non-null  int64
```

```
 4   Dependents                  52000 non-null  int64
 5   Education                   52000 non-null  object
 6   Employment_Status           52000 non-null  object
 7   Occupation_Type             52000 non-null  object
 8   Residential_Status          52000 non-null  object
 9   City/Town                   52000 non-null  object
 10  Annual_Income               52000 non-null  int64
 11  Monthly_Expenses            52000 non-null  int64
 12  Credit_Score                52000 non-null  int64
 13  Existing_Loans              52000 non-null  int64
 14  Total_Existing_Loan_Amount  52000 non-null  int64
 15  Outstanding_Debt            52000 non-null  int64
 16  Loan_History                52000 non-null  int64
 17  Loan_Amount_Requested       52000 non-null  int64
 18  Loan_Term                   52000 non-null  int64
 19  Loan_Purpose                52000 non-null  object
 20  Interest_Rate               52000 non-null  float64
 21  Loan_Type                   52000 non-null  object
 22  Co-Applicant                52000 non-null  object
 23  Bank_Account_History        52000 non-null  int64
 24  Transaction_Frequency       52000 non-null  int64
 25  Default_Risk                52000 non-null  float64
 26  Loan_Approval_Status        52000 non-null  int64
dtypes: float64(2), int64(15), object(10)
memory usage: 10.7+ MB
```

[ ]:
```
#DESCRIPTIVE ANALYSIS
```

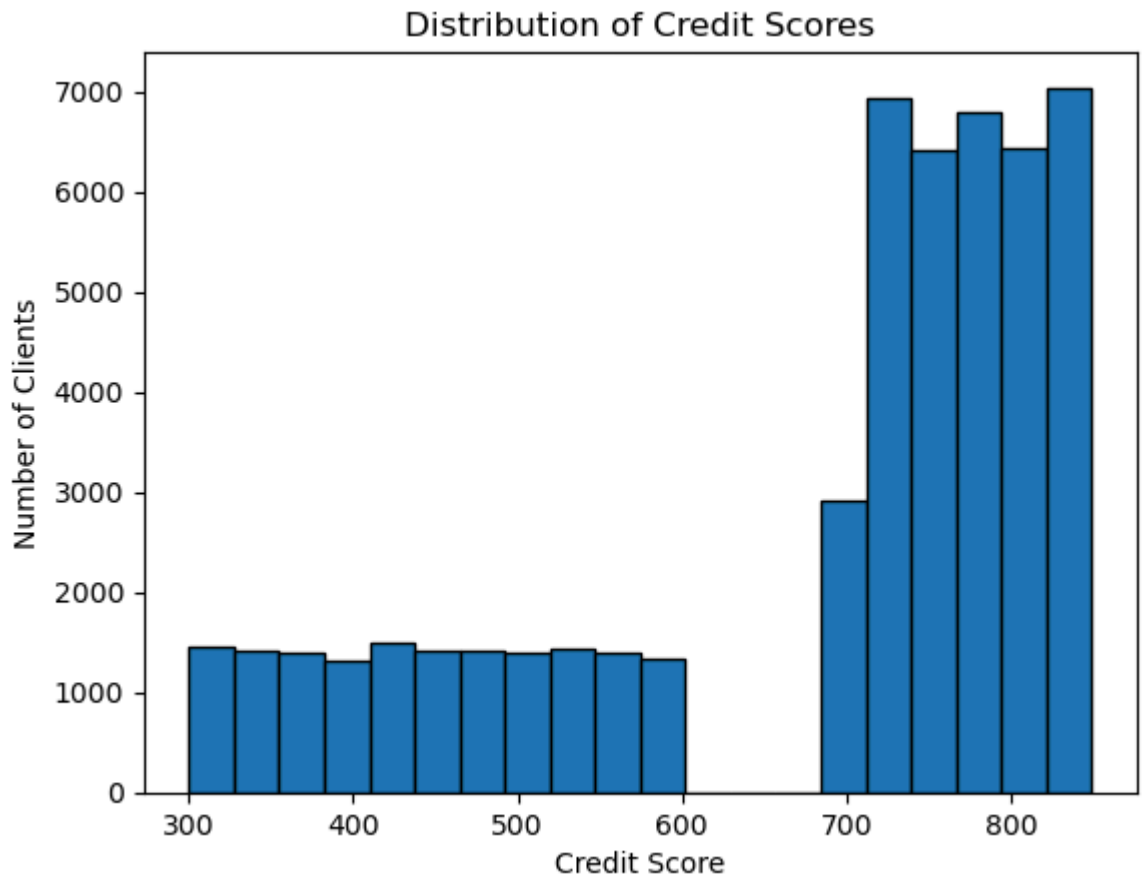
[21]:
```
print("\n--- Summary Statistics ---")
print(df1["Credit_Score"].describe())
```

```
--- Summary Statistics ---
count    52000.000000
mean       678.089019
std        159.990367
min        300.000000
25%        553.000000
50%        742.000000
75%        796.000000
max        849.000000
Name: Credit_Score, dtype: float64
```

[ ]:
```
#Distribution plot
```

[29]:
```
plt.hist(df1["Credit_Score"], bins = 20, edgecolor ='black')
```

[29]:
```
plt.hist(df1["Credit_Score"], bins = 20, edgecolor ='black')
plt.title("Distribution of Credit Scores")
plt.xlabel("Credit Score")
plt.ylabel("Number of Clients")
plt.show()
```



[ ]:
```
d 1500 clients fall between 300 and 600 credit score, while on the other hand, around 3000 and above clients fall between more tha
```

[ ]:
```
#SEGMENTATION OF CLIENT INTO SCORE BAND
```

[37]:
```
bins = [0, 300, 500, 700, 900]
labels = ['High Risk', 'Medium Risk', 'Low Risk', 'Very Low Risk']
df1['score_band'] = pd.cut(df1['Credit_Score'], bins=bins, labels=labels)
```

[39]:
```
df1['score_band']
```

[37]:
```
bins = [0, 300, 500, 700, 900]
labels = ['High Risk', 'Medium Risk', 'Low Risk', 'Very Low Risk']
df1['score_band'] = pd.cut(df1['Credit_Score'], bins=bins, labels=labels)
```

[39]:
```
df1['score_band']
```

[39]:
```
0        Very Low Risk
1          Medium Risk
2          Medium Risk
3        Very Low Risk
4        Very Low Risk
             ...
51995      Medium Risk
51996    Very Low Risk
51997      Medium Risk
51998      Medium Risk
51999    Very Low Risk
Name: score_band, Length: 52000, dtype: category
Categories (4, object): ['High Risk' < 'Medium Risk' < 'Low Risk' < 'Very Low Risk']
```

[49]:
```
print(df1["Loan_History"].describe())
```

```
count    52000.000000
mean         0.198596
std          0.398947
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max          1.000000
Name: Loan_History, dtype: float64
```

[ ]:
```
#DEFAULT RATE BY SCORE BAND
```

[59]:
```
default_rate = df1.groupby('score_band')["Loan_History"].mean().reset_index()
print("\n--- Default Rate By Score Band ---")
print(default_rate)
```

```
--- Default Rate By Score Band ---
      score_band  Loan_History
0      High Risk      0.181818
1    Medium Risk      0.200972
2       Low Risk      0.194047
3  Very Low Risk      0.198622
```

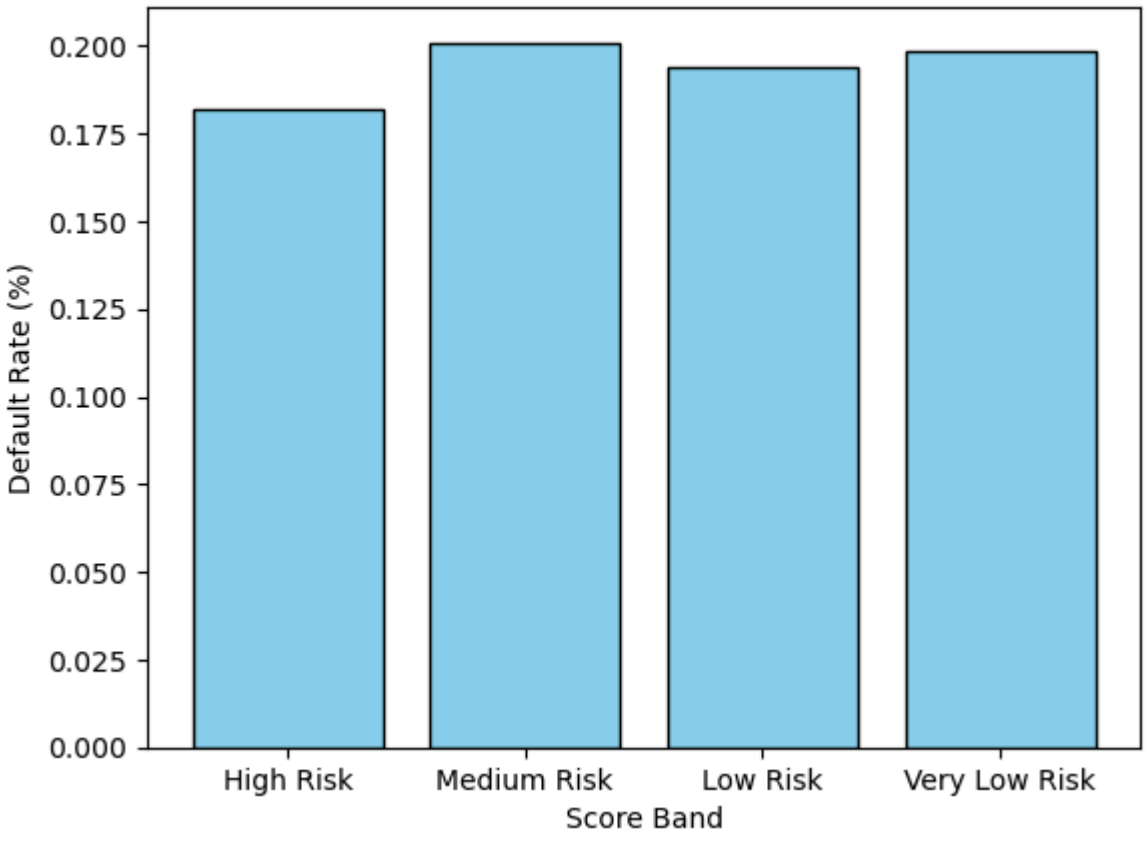
```
0      High Risk      0.181818
1    Medium Risk      0.200972
2       Low Risk      0.194047
3  Very Low Risk      0.198622
```

```
C:\Users\LENOVO\AppData\Local\Temp\ipykernel_3344\2684578765.py:1: FutureWarning: The default of observed=False is deprecated and
in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and si
  default_rate = df1.groupby('score_band')["Loan_History"].mean().reset_index()
```

[ ]:
```
# THE ABOVE MEANS 18% OF CLIENTS IN THE HIGH RISK  BAND DEFAULTED
```

[ ]:
```
# bAR CHART FOR DEFAULT RATE
```

[61]:
```
plt.bar(default_rate["score_band"], default_rate["Loan_History"], color='skyblue',edgecolor='black')
plt.title("Default Rate by Credit Score Band")
plt.xlabel("Score Band")
plt.ylabel("Default Rate (%)")
plt.show()
```



[ ]:
```
#Model Performance Analysis
```

[63]:
```
auc = roc_auc_score(df1["Loan_History"], df1["Credit_Score"])
print(f"\nAUC  (Discriminatory Power):{auc:.3f}")
```

```
AUC  (Discriminatory Power):0.498
```

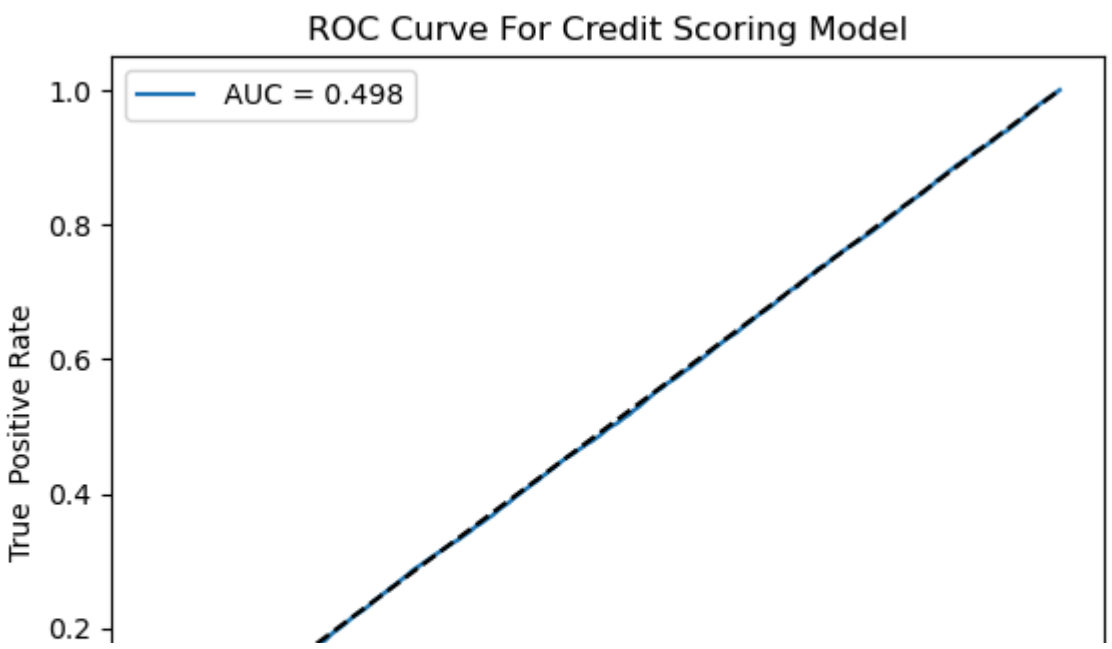
[ ]:
```
#KS statistics
```

[69]:
```
fpr, tpr, thresholds = roc_curve(df1["Loan_History"],df1["Credit_Score"])
ks_statistic = max(tpr - fpr)
print(f"KS Statistic:{ks_statistic:.3f}")
```

```
KS Statistic:0.003
```

[ ]:
```
#plot ROC curve
```

[71]:
```
plt.plot(fpr, tpr, label=f' AUC = {auc:.3f}')
plt.plot([0,1], [0,1], 'k--')
plt.xlabel("False Positive Rate")
plt.ylabel("True  Positive Rate ")
plt.title("ROC Curve For Credit Scoring Model")
plt.legend()
plt.show()
```



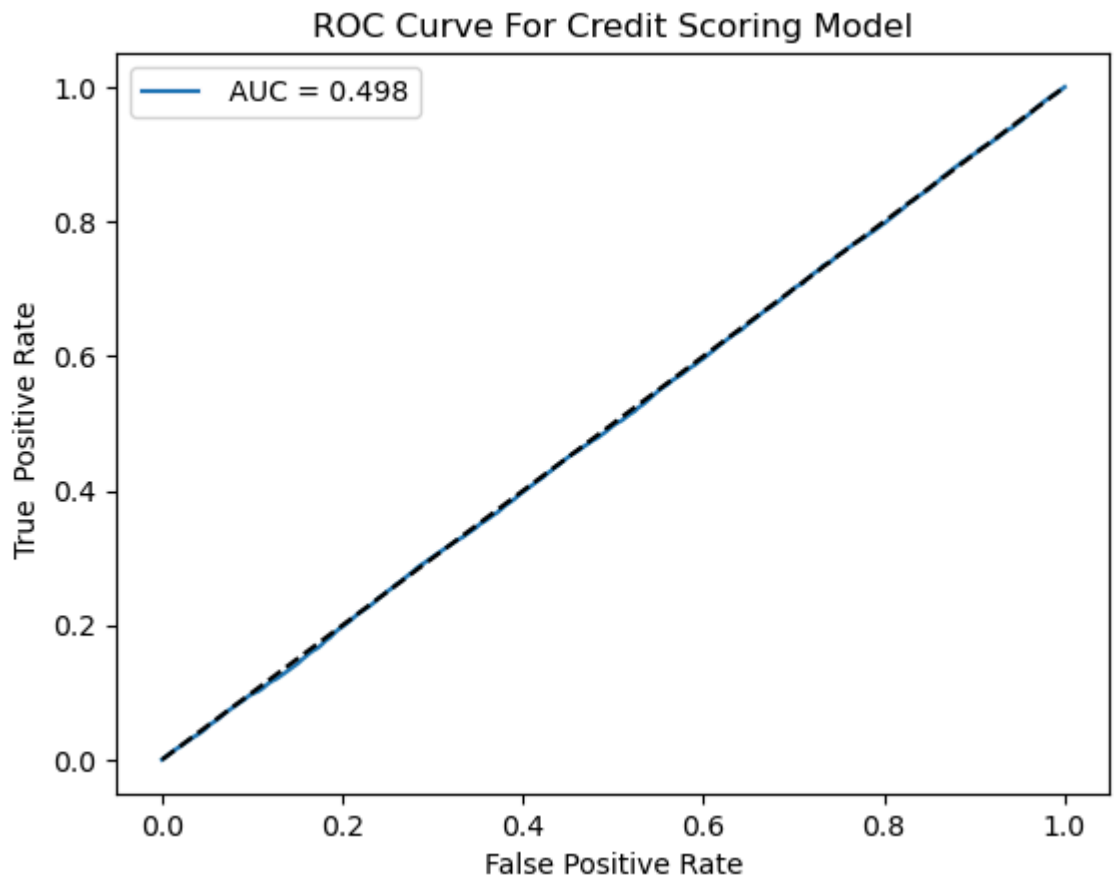
jupyter Untitled7 Last Checkpoint: now

File Edit View Run Kernel Settings Help

Code JupyterLab Pytho

[ ]:
```
#plot ROC curve
```

[71]:
```
plt.plot(fpr, tpr, label=f' AUC = {auc:.3f}')
plt.plot([0,1], [0,1], 'k--')
plt.xlabel("False Positive Rate")
plt.ylabel("True  Positive Rate ")
plt.title("ROC Curve For Credit Scoring Model")
plt.legend()
plt.show()
```



[ ]: