airk8vzg6

August 2, 2023

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

[3]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[4]:
```
df_train=pd.read_csv("/content/drive/MyDrive/mydatasets/C8_loan-train.csv")
df_test=pd.read_csv("/content/drive/MyDrive/mydatasets/C8_loan-test.csv")
```

[5]:
```
df_train.dropna(inplace=True)
df_test.dropna(inplace=True)
```

[6]:
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 480 entries, 1 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            480 non-null    object
 1   Gender             480 non-null    object
 2   Married            480 non-null    object
 3   Dependents         480 non-null    object
 4   Education          480 non-null    object
 5   Self_Employed      480 non-null    object
 6   ApplicantIncome    480 non-null    int64
 7   CoapplicantIncome  480 non-null    float64
 8   LoanAmount         480 non-null    float64
 9   Loan_Amount_Term   480 non-null    float64
 10  Credit_History     480 non-null    float64
 11  Property_Area      480 non-null    object
 12  Loan_Status        480 non-null    object
```

```
dtypes: float64(4), int64(1), object(8)
memory usage: 52.5+ KB
```

```
[7]: df_train.columns
```

```
[7]: Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
           'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
           'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'],
          dtype='object')
```

```
[8]: feature_matrix = df_train[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
            'Loan_Amount_Term', 'Credit_History']]
     target_vector = df_train[['Self_Employed']]
```

```
[9]: fs = StandardScaler().fit_transform(feature_matrix)
     logr = LogisticRegression()
     logr.fit(fs,target_vector)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

```
[9]: LogisticRegression()
```

```
[10]: observation = df_test[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
             'Loan_Amount_Term', 'Credit_History']]
      prediction = logr.predict(observation)
      print(prediction)
```

```
['Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'No' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes' 'Yes'
 'Yes']
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has
feature names, but LogisticRegression was fitted without feature names
  warnings.warn(
```

[11]:
```
logr.classes_
```

[11]:
```
array(['No', 'Yes'], dtype=object)
```

[12]:
```
logr.predict_proba(observation)[0][0]
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:432: UserWarning: X has
feature names, but LogisticRegression was fitted without feature names
  warnings.warn(
```

[12]:
```
0.0
```

Random Forest

[22]:
```
df2=pd.read_csv("/content/drive/MyDrive/mydatasets/C8_loan-test.csv")
df2.dropna(inplace=True)
df2['Self_Employed'].value_counts()
```

[22]:
```
No     257
Yes     32
Name: Self_Employed, dtype: int64
```

[23]:
```
x=df2[['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term', 'Credit_History']]
y=df2['Self_Employed']
```

[24]:
```
g1={'Self_Employed':{'No':1, 'Yes':2}}
df2=df2.replace(g1)
df2
```

[24]:
```
    Loan_ID Gender Married Dependents     Education  Self_Employed  \
0  LP001015   Male     Yes          0      Graduate              1
1  LP001022   Male     Yes          1      Graduate              1
2  LP001031   Male     Yes          2      Graduate              1
4  LP001051   Male      No          0  Not Graduate              1
5  LP001054   Male     Yes          0  Not Graduate              2
```

```
..        …     …      …         …           …           …
361  LP002969   Male     Yes         1      Graduate             1
362  LP002971   Male     Yes        3+  Not Graduate             2
363  LP002975   Male     Yes         0      Graduate             1
365  LP002986   Male     Yes         0      Graduate             1
366  LP002989   Male      No         0      Graduate             2

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0               5720                  0       110.0             360.0
1               3076               1500       126.0             360.0
2               5000               1800       208.0             360.0
4               3276                  0        78.0             360.0
5               2165               3422       152.0             360.0
..               …                  …          …                 …
361             2269               2167        99.0             360.0
362             4009               1777       113.0             360.0
363             4158                709       115.0             360.0
365             5000               2393       158.0             360.0
366             9200                  0        98.0             180.0

     Credit_History Property_Area
0               1.0         Urban
1               1.0         Urban
2               1.0         Urban
4               1.0         Urban
5               1.0         Urban
..              …           …
361             1.0     Semiurban
362             1.0         Urban
363             1.0         Urban
365             1.0         Rural
366             1.0         Rural

[289 rows x 12 columns]
```

[25]:
```
from sklearn.model_selection import train_test_split
```

[26]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

[27]:
```
from sklearn.ensemble import RandomForestClassifier
```

[28]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

[28]:
```
RandomForestClassifier()
```

[29]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]
}
```

[30]:
```
from sklearn.model_selection import GridSearchCV
grid_search␣
 ↪=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

[30]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

[31]:
```
grid_search.best_score_
```

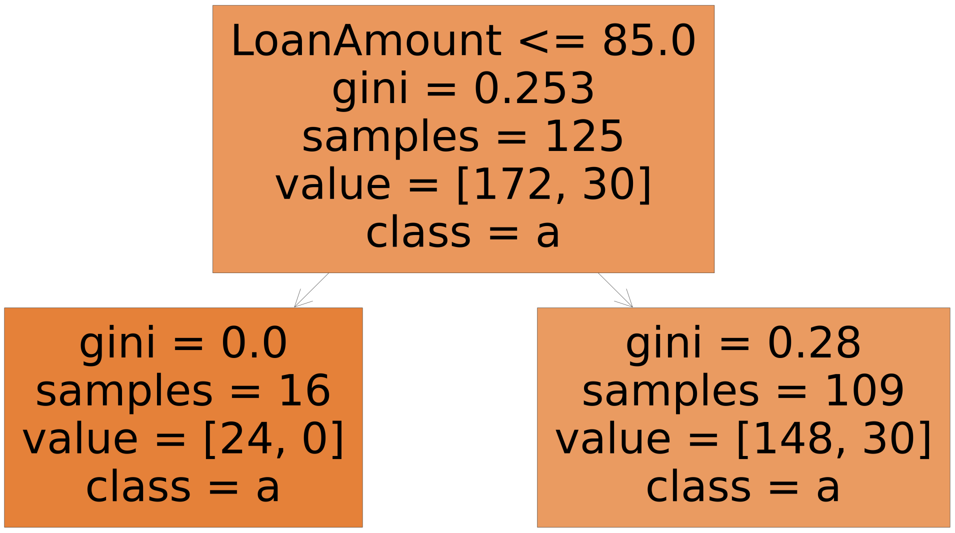
[31]:
```
0.8762376237623762
```

[32]:
```
rfc_best=grid_search.best_estimator_
```

[33]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.
 ↪columns,class_names=['a','b'],filled=True)
```

[33]:
```
[Text(0.5, 0.75, 'LoanAmount <= 85.0\ngini = 0.253\nsamples = 125\nvalue = [172,
30]\nclass = a'),
 Text(0.25, 0.25, 'gini = 0.0\nsamples = 16\nvalue = [24, 0]\nclass = a'),
 Text(0.75, 0.25, 'gini = 0.28\nsamples = 109\nvalue = [148, 30]\nclass = a')]
```


LoanAmount <= 85.0
gini = 0.253
samples = 125
value = [172, 30]
class = a
gini = 0.0
samples = 16
value = [24, 0]
class = a
gini = 0.28
samples = 109
value = [148, 30]
class = a