Project 3: Build Decision Tree for Bank Loan Modeling

DV - "Personal Loan"

IDV - Output of RF Algorithm

Step 1: Load the dataset

import pandas as pd

dataset1 = pd.read\_excel("D:/AI\_ML\_Course/Day21/dataset/Bank\_Personal\_Loan\_Modelling.xlsx", sheet\_name='Data')

dataset1.columns

Out[8]:

Index(['ID', 'Age', 'Experience', 'Income', 'ZIP Code', 'Family', 'CCAvg',

'Education', 'Mortgage', 'Personal Loan', 'Securities Account',

'CD Account', 'Online', 'CreditCard'],

dtype='object')

dataset2 = dataset1.drop('ID', axis=1)

dataset3 = dataset2.drop('ZIP Code', axis=1)

dataset3.describe()

Out[9]:

Age Experience ... Online CreditCard

count 5000.000000 5000.000000 ... 5000.000000 5000.000000

mean 45.338400 20.104600 ... 0.596800 0.294000

std 11.463166 11.467954 ... 0.490589 0.455637

min 23.000000 -3.000000 ... 0.000000 0.000000

25% 35.000000 10.000000 ... 0.000000 0.000000

50% 45.000000 20.000000 ... 1.000000 0.000000

75% 55.000000 30.000000 ... 1.000000 1.000000

max 67.000000 43.000000 ... 1.000000 1.000000

dataset3["Income"].isnull().sum()

Out[20]: 0

dataset3["CCAvg"].isnull().sum()

Out[21]: 0

dataset3["Education"].isnull().sum()

Out[22]: 0

Step 2: Use the Random forest Algorithm to find the important features.

from sklearn import tree

from sklearn import preprocessing

from sklearn.ensemble import RandomForestClassifier

label\_encoder = preprocessing.LabelEncoder()

rf\_model = RandomForestClassifier(n\_estimators = 1000, max\_features = 2 , oob\_score = True)

features = ['Age', 'Experience', 'Income', 'Family', 'CCAvg','Education', 'Mortgage','Securities Account','CD Account', 'Online', 'CreditCard']

rf\_model.fit(X=dataset3[features],y=dataset3["Personal Loan"])

print("OOB Accuracy:")

OOB Accuracy:

print(rf\_model.oob\_score\_);

0.988

for feature, imp in zip(features, rf\_model.feature\_importances\_):

print(feature, imp)

Age 0.04489275350455519

Experience 0.04394514026471026

Income 0.346879001014009

Family 0.0984189891492654

CCAvg 0.18418828162811646

Education 0.1621605380003643

Mortgage 0.043584162094948094

Securities Account 0.005355082670786246

CD Account 0.05243427196261527

Online 0.008220612725583542

CreditCard 0.00992116698504641

We observe with above values from Random forest algorithm that **Income, CCAvg & Education features** should be used for Decision Tree model as they have P values more than 0.5

Step 3: Create the Decision Tree with important features (Independent variable) as Income, CCAvg & Education and Dependent variable as Personal loan

test\_features = pd.DataFrame([dataset3["Income"],dataset3["CCAvg"],dataset3["Education"]]).T

tree\_model = tree.DecisionTreeClassifier()

tree\_model.fit(X=test\_features, y=dataset3['Personal Loan'])

Out[56]:

DecisionTreeClassifier(ccp\_alpha=0.0, class\_weight=None, criterion='gini',

max\_depth=None, max\_features=None, max\_leaf\_nodes=None,

min\_impurity\_decrease=0.0, min\_impurity\_split=None,

min\_samples\_leaf=1, min\_samples\_split=2,

min\_weight\_fraction\_leaf=0.0, presort='deprecated',

random\_state=None, splitter='best')

with open("D:/AI\_ML\_Course/Day24/Dtree.dot",'w') as f:

f = tree.export\_graphviz(tree\_model,feature\_names=["Income","CCAvg","Education"],out\_file=f);



To reduce the complexity, set the max\_depth as 6:

test\_features = pd.DataFrame([dataset3["Income"],dataset3["CCAvg"],dataset3["Education"]]).T

tree\_model = tree.DecisionTreeClassifier(max\_depth=6)

tree\_model.fit(X=test\_features,y=dataset3['Personal Loan'])

with open("D:/Growth/AI\_ML\_Course/Day24/Dtree0.dot",'w') as f:

f = tree.export\_graphviz(tree\_model,feature\_names=["Income","CCAvg","Education"],out\_file=f);



Step 4: To calculate the accuracy of the model

tree\_model.score(X=test\_features, y=dataset3['Personal Loan'])

Out[7]: 0.9972

Inference: The model is 99.72% accurate

After reducing the complexity/setting max\_depth as 6,

tree\_model.score(X=test\_features,y=dataset3['Personal Loan'])

Out[23]: 0.9738

Inference: The model is 97.38% accurate

Step 5: Based on the training Dataset predicted the test dataset output

test\_preds = tree\_model.predict(X=test\_features)

predicted\_output =pd.DataFrame({"ID":dataset1['ID'],'Personal Loan':test\_preds})

predicted\_output.to\_csv("D:/AI\_ML\_Course/Day24/OutputBM.csv", index=False)



Step 6: Rules based on Decision Tree

* Out of 5000 Samples, there are 4520 people with Income less than $113,000 and 480 people with greater than $113,000.
* Further 4021 Samples divided as per CCAvg having weighted mean as 2.95, where 3937 having less than weighted mean
* Further 979 Samples divided as per Education having weighted mean as 1.5, where 583 having less than weighted mean.