PROJECT 3: Build Decision Tree for Bank Loan Modelling

DV - "Personal Loan"

IDV - Output of RF Algorithm

bank\_loan=pd.read\_excel("Bank\_Personal\_Loan\_Modelling.xlsx",sheet\_name=1)

bank\_loan.head()

Out[17]:

ID Age Experience ... CD Account Online CreditCard

0 1 25 1 ... 0 0 0

1 2 45 19 ... 0 0 0

2 3 39 15 ... 0 0 0

3 4 35 9 ... 0 0 0

4 5 35 8 ... 0 0 1

[5 rows x 14 columns]

bank\_loan.isnull().sum()

Out[18]:

ID 0

Age 0

Experience 0

Income 0

ZIP Code 0

Family 0

CCAvg 0

Education 0

Mortgage 0

Personal Loan 0

Securities Account 0

CD Account 0

Online 0

CreditCard 0

dtype: int64

bank\_loan.isna().sum().sum

Out[19]:

<bound method Series.sum of ID 0

Age 0

Experience 0

Income 0

ZIP Code 0

Family 0

CCAvg 0

Education 0

Mortgage 0

Personal Loan 0

Securities Account 0

CD Account 0

Online 0

CreditCard 0

dtype: int64>

from sklearn.ensemble import RandomForestClassifier

bank\_loan.columns

Out[21]:

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

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

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

dtype='object')

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

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

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

Out[26]:

RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None,

criterion='gini', max\_depth=None, max\_features=2,

max\_leaf\_nodes=None, max\_samples=None,

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

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

min\_weight\_fraction\_leaf=0.0, n\_estimators=1000,

n\_jobs=None, oob\_score=True, random\_state=None,

verbose=0, warm\_start=False)

print("OOB Accuracy:",rf\_model.oob\_score\_)

OOB Accuracy: 0.9878

**OOB Accuracy IS 98.78%**

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

print(feature,imp);

Age 0.04543651167894422

Experience 0.04543253449777488

Income 0.34643552431738867

Family 0.09354154231066233

CCAvg 0.18372204079891896

Mortgage 0.044985356045961086

Securities Account 0.005464178092783381

CD Account 0.055314547163470194

Online 0.008617824910224867

CreditCard 0.009973615133821397

Education 0.16107632505005004

**The important variables for decision tree are**

**1.Income**

**2.CCAvg**

**3.Education**

tree\_model=tree.DecisionTreeClassifier()

predictors=pd.DataFrame([bank\_loan["Income"],bank\_loan["CCAvg"],bank\_loan["Education"]]).T

tree\_model.fit(X=predictors,y=bank\_loan["Personal Loan"])

Out[32]:

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("BankDtree.dot",'w') as f:

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

tree\_model.score(X=predictors,y=bank\_loan["Personal Loan"])

Out[34]: 0.9972

tree\_model=tree.DecisionTreeClassifier(max\_depth=8)

predictors=pd.DataFrame([bank\_loan["Income"],bank\_loan["CCAvg"],bank\_loan["Education"]]).T

tree\_model.fit(X=predictors,y=bank\_loan["Personal Loan"])

Out[37]:

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

max\_depth=8, 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("DtreeBank.dot",'w') as f:

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