Bank Loan Analysis

```
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn import tree
```

Loading Data and Data Treatment:

```
loan_data = pd.read_excel("Bank_Personal_Loan_Modelling.xlsx", sheet_name= "Data")

loan_data.head(2)
Out[6]:
   ID  Age  Experience  ...  CD Account  Online  CreditCard
0   1   25           1  ...           0       0           0
1   2   45          19  ...           0       0           0

[2 rows x 14 columns]

loan_data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   ID                  5000 non-null   int64
 1   Age                 5000 non-null   int64
 2   Experience          5000 non-null   int64
 3   Income              5000 non-null   int64
 4   ZIP Code            5000 non-null   int64
 5   Family              5000 non-null   int64
 6   CCAvg               5000 non-null   float64
```

```
 7   Education          5000 non-null   int64
 8   Mortgage           5000 non-null   int64
 9   Personal Loan      5000 non-null   int64
 10  Securities Account  5000 non-null   int64
 11  CD Account         5000 non-null   int64
 12  Online             5000 non-null   int64
 13  CreditCard         5000 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 547.0 KB

loan_data.isna().sum()
Out[8]:
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

loan_data.columns
Out[9]:
Index(['ID', 'Age', 'Experience', 'Income', 'ZIP Code', 'Family', 'CCAvg',
```

```
      'Education', 'Mortgage', 'Personal Loan', 'Securities Account',
      'CD Account', 'Online', 'CreditCard'],
     dtype='object')
```

Random Forest Algorithm to find imp Variables

```
from sklearn.ensemble import RandomForestClassifier

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

rf_model = RandomForestClassifier(n_estimators= 1000, max_features= 2, oob_score= True)

rf_model.fit(X= loan_data[features], y = loan_data['Personal Loan'])
Out[13]:
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("RF_Model Accuracy:", rf_model.oob_score_)
```

RF_Model Accuracy: 0.9872

```
for fetaure,imp in zip(features,rf_model.feature_importances_):
   print(fetaure,imp)
```

Age 0.0448617731716443

Experience 0.04458422429350977

Income 0.3447982578131703

Family 0.09650893727430132

CCAvg 0.18408847848020293

Education 0.1628971002549275

Mortgage 0.043677061361758356

Securities Account 0.005347821587452683

CD Account 0.05458233762645788

Online 0.008596153360164793

CreditCard 0.010057854776410264

Generating Decision Tree Model

```
predictors = loan_data[['Income','Family','CCAvg','Education']]


tree_model = tree.DecisionTreeClassifier(max_depth= 8, max_leaf_nodes= 10)


tree_model.fit(X= predictors, y = loan_data['Personal Loan'])
```

Out[21]:

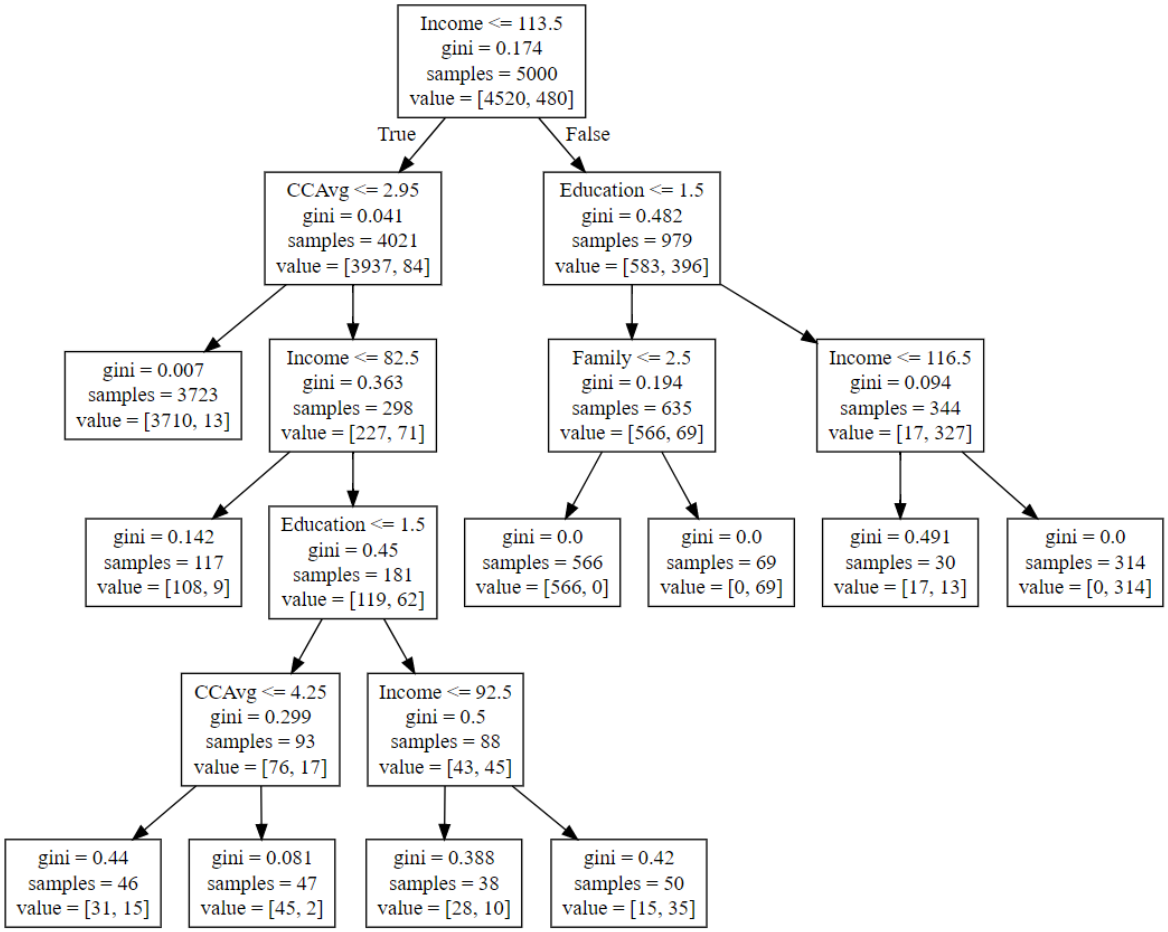
```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=8, max_features=None, max_leaf_nodes=10,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')


with open("Loan_Dtree1.dot","w") as f:
   f = tree.export_graphviz(tree_model, feature_names=['Income','Family','CCAvg','Education'],
out_file= f)
```

print("DTree Model Accuracy:", tree_model.score(X= predictors, y = loan_data['Personal Loan']))

DTree Model Accuracy: 0.9846

Decision Tree:



Rules:

LOAN - NO

1. If the CCAvg is less than 2.95 and the Income is less than 113.5, then probability of Loan(No) is high
2. If the CCAvg is greater than 2.95 and the Income is less than 82.5, then probability of Loan(No) is high
3. If Education is less than 1.5, CCAvg is in the range of 2.95 to 4.25 and Income is in the range of 82.5 to 113.5, then probability of Loan(No) is high
4. If Education is less than 1.5, CCAvg is greater than 4.25 and Income is in the range of 82.5 to 113.5, then probability of Loan(No) is high
5. If Education is greater than 1.5, CCAvg is greater than 2.95 and Income is in the range of 82.5 to 92.5, then probability of Loan(No) is high
6. If Income is greater than 113.5, Education is less than 1.5 and Family less than 2.5, then probability of Loan(No) is high

LOAN - YES

1. If Education is greater than 1.5, CCAvg is greater than 2.95 and Income is in the range of 92.5 to 113.5, then probability of Loan(Yes) is high
2. If Income is greater than 113.5, Education is less than 1.5 and Family greater than 2.5, then probability of Loan(Yes) is high
3. If Income is in range of 113.5 to 116.5 and Education is greater than 1.5, then probability of Loan(Yes) is almost equal
4. If Income is greater than 116.5, Education is greater than 1.5, then probability of Loan(Yes) is high

Inference:

1. Based on the importance value generated with Random forest algorithm, it is seen that the features **'Income', 'Family', 'CCAvg' and 'Education'** are more significant for decision tree generation.
2. Decision tree generated with these features and max-depth of 8 and 10 leaf nodes provides **98.46%** accuracy in classifying the record as Personal Loan(Y/N)