**Project 1: Build Decision Tree(DV-"Survived",IDV-"Age,Gender and Fare") and Prediction**

**Decision Tree:**

#import packages

import pandas as pd

import numpy as np

from sklearn import tree

from sklearn import preprocessing

#load training dataset

titanic\_train=pd.read\_csv("train.csv")

titanic\_train=titanic\_train.drop('Name',axis=1)

titanic\_train=titanic\_train.drop('PassengerId',axis=1)

titanic\_train=titanic\_train.drop('Pclass',axis=1)

titanic\_train=titanic\_train.drop('SibSp',axis=1)

titanic\_train=titanic\_train.drop('Parch',axis=1)

titanic\_train=titanic\_train.drop('Ticket',axis=1)

titanic\_train=titanic\_train.drop('Cabin',axis=1)

titanic\_train=titanic\_train.drop('Embarked',axis=1)

print(titanic\_train.isna().sum())

#convert string values to numerical

labelencoder=preprocessing.LabelEncoder()

encoded\_sex=labelencoder.fit\_transform(titanic\_train["Sex"])

#make the model

tree\_model=tree.DecisionTreeClassifier(max\_depth=4)

predictors=pd.DataFrame([encoded\_sex,titanic\_train["Age"],titanic\_train["Fare"]]).T

tree\_model.fit(X=predictors,y=titanic\_train["Survived"])

with open("Titanic\_DTree.dot",'w') as f:

f=tree.export\_graphviz(tree\_model,feature\_names=["Sex","Age","Fare"],out\_file=f);

#Prediction-load test data

titanic\_test=pd.read\_csv("test.csv")

#remove unimportant variables

titanic\_test=titanic\_test.drop('Name',axis=1)

titanic\_test=titanic\_test.drop('Pclass',axis=1)

titanic\_test=titanic\_test.drop('SibSp',axis=1)

titanic\_test=titanic\_test.drop('Parch',axis=1)

titanic\_test=titanic\_test.drop('Ticket',axis=1)

titanic\_test=titanic\_test.drop('Embarked',axis=1)

print(titanic\_test.isnull().sum())

#convert string values to numerical

encoded\_sex\_test=labelencoder.fit\_transform(titanic\_test["Sex"])

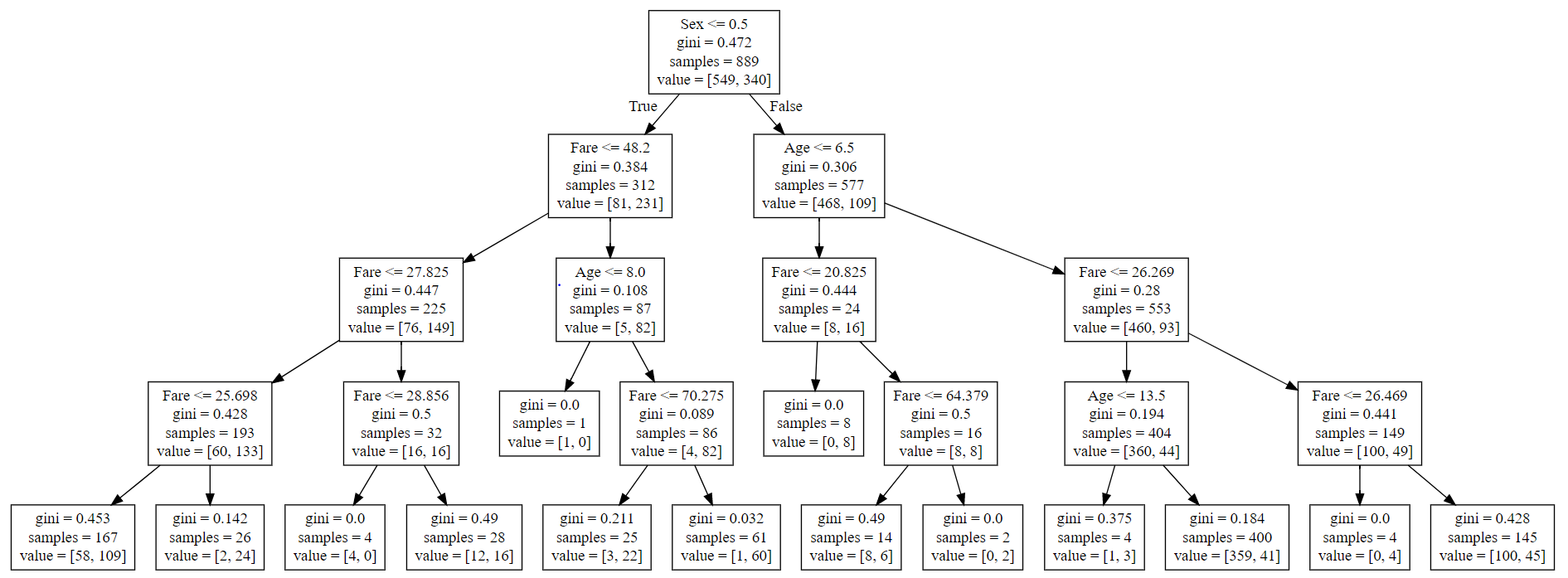
predictors\_test=pd.DataFrame([encoded\_sex\_test,titanic\_test["Age"],titanic\_test["Fare"]]).T

test\_preds=tree\_model.predict(X=predictors\_test)

predicted\_output=pd.DataFrame({"PassengerId":titanic\_test["PassengerId"],"Survived":test\_preds})

predicted\_output.to\_csv("Titanic\_DTree\_Predicted.csv",index=False);

**Output:**



**Project 2: Build Decision Tree for Attrition Rate Analysis**

**DV - "Attrition"**

**IDV - Output of RF Algorithm**

**RF Algo:**

#import packages

from sklearn.ensemble import RandomForestClassifier

import pandas as pd

import numpy as np

from sklearn import preprocessing

#load dataset

dataset=pd.read\_csv("general\_data.csv")

#converting string variables into Numerical

le=preprocessing.LabelEncoder()

le.fit(dataset["Gender"])

dataset["Gender"]=le.transform(dataset["Gender"])

le.fit(dataset["Attrition"])

dataset["Attrition"]=le.transform(dataset["Attrition"])

le.fit(dataset["BusinessTravel"])

dataset["BusinessTravel"]=le.transform(dataset["BusinessTravel"])

le.fit(dataset["Department"])

dataset["Department"]=le.transform(dataset["Department"])

le.fit(dataset["EducationField"])

dataset["EducationField"]=le.transform(dataset["EducationField"])

le.fit(dataset["JobRole"])

dataset["JobRole"]=le.transform(dataset["JobRole"])

le.fit(dataset["MaritalStatus"])

dataset["MaritalStatus"]=le.transform(dataset["MaritalStatus"])

le.fit(dataset["MaritalStatus"])

dataset["MaritalStatus"]=le.transform(dataset["MaritalStatus"])

le.fit(dataset["Over18"])

dataset["Over18"]=le.transform(dataset["Over18"])

#checking for null values

print(dataset.isna().sum())

#replacing null values with average of variable

print("mean of NumCompaniesWorked: ",dataset["NumCompaniesWorked"].mean())

new\_NCW\_var=np.where(dataset["NumCompaniesWorked"].isnull(),2,dataset["NumCompaniesWorked"])

dataset["NumCompaniesWorked"]=new\_NCW\_var

print("mean of TotalWorkingYears: ",dataset["TotalWorkingYears"].mean())

new\_TWY\_var=np.where(dataset["TotalWorkingYears"].isnull(),11,dataset["TotalWorkingYears"])

dataset["TotalWorkingYears"]=new\_TWY\_var

#recheck for null values

print(dataset.isna().sum())

#build model

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

features=["Age","BusinessTravel", "Department", "DistanceFromHome","Education", "EducationField", "EmployeeCount", "EmployeeID", "Gender","JobLevel", "JobRole", "MaritalStatus", "MonthlyIncome","NumCompaniesWorked", "Over18", "PercentSalaryHike", "StandardHours","StockOptionLevel", "TotalWorkingYears", "TrainingTimesLastYear","YearsAtCompany", "YearsSinceLastPromotion", "YearsWithCurrManager"]

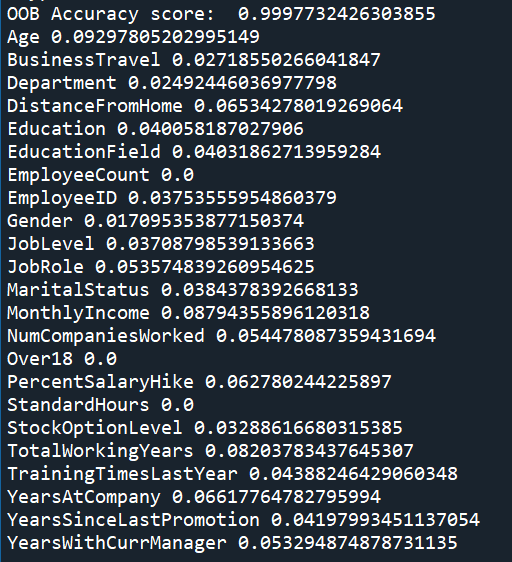
rf\_model.fit(X=dataset[features],y=dataset["Attrition"])

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

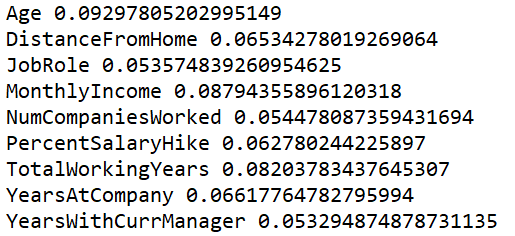
#find important variable

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

print(features,imp)

**RF Output:**

Values>0.05 :



* **IDV:** Age DistanceFromHome JobRole MonthlyIncome NumCompaniesWorked PercentSalaryHike TotalWorkingYears YearsAtCompany YearsWithCurrManager are important variable considered for Decition Tree Algorithm

**Decision Tree:**

#import packages

import pandas as pd

import numpy as np

from sklearn import tree

from sklearn import preprocessing

#load dataset dataset

attrition\_dataset=pd.read\_csv("general\_data.csv")

attrition\_dataset=attrition\_dataset.drop('BusinessTravel',axis=1)

attrition\_dataset=attrition\_dataset.drop('Department',axis=1)

attrition\_dataset=attrition\_dataset.drop('Education',axis=1)

attrition\_dataset=attrition\_dataset.drop('EducationField',axis=1)

attrition\_dataset=attrition\_dataset.drop('EmployeeCount',axis=1)

attrition\_dataset=attrition\_dataset.drop('EmployeeID',axis=1)

attrition\_dataset=attrition\_dataset.drop('Gender',axis=1)

attrition\_dataset=attrition\_dataset.drop('JobLevel',axis=1)

attrition\_dataset=attrition\_dataset.drop('MaritalStatus',axis=1)

attrition\_dataset=attrition\_dataset.drop('Over18',axis=1)

attrition\_dataset=attrition\_dataset.drop('StandardHours',axis=1)

attrition\_dataset=attrition\_dataset.drop('StockOptionLevel',axis=1)

attrition\_dataset=attrition\_dataset.drop('TrainingTimesLastYear',axis=1)

attrition\_dataset=attrition\_dataset.drop('YearsSinceLastPromotion',axis=1)

print(attrition\_dataset.isna().sum())

#replacing null values with average of variable

print("mean of NumCompaniesWorked: ",attrition\_dataset["NumCompaniesWorked"].mean())

new\_NCW\_var=np.where(attrition\_dataset["NumCompaniesWorked"].isnull(),2,attrition\_dataset["NumCompaniesWorked"])

attrition\_dataset["NumCompaniesWorked"]=new\_NCW\_var

print("mean of TotalWorkingYears: ",attrition\_dataset["TotalWorkingYears"].mean())

new\_TWY\_var=np.where(attrition\_dataset["TotalWorkingYears"].isnull(),11,attrition\_dataset["TotalWorkingYears"])

attrition\_dataset["TotalWorkingYears"]=new\_TWY\_var

print(attrition\_dataset.isna().sum())

#convert string values to numerical

labelencoder=preprocessing.LabelEncoder()

encoded\_JobRole=labelencoder.fit\_transform(attrition\_dataset["JobRole"])

#make the model

predictors=pd.DataFrame([encoded\_JobRole,attrition\_dataset["Age"],attrition\_dataset["DistanceFromHome"],attrition\_dataset["MonthlyIncome"],attrition\_dataset["PercentSalaryHike"],attrition\_dataset["NumCompaniesWorked"],attrition\_dataset["TotalWorkingYears"],attrition\_dataset["YearsAtCompany"],attrition\_dataset["YearsWithCurrManager"]]).T

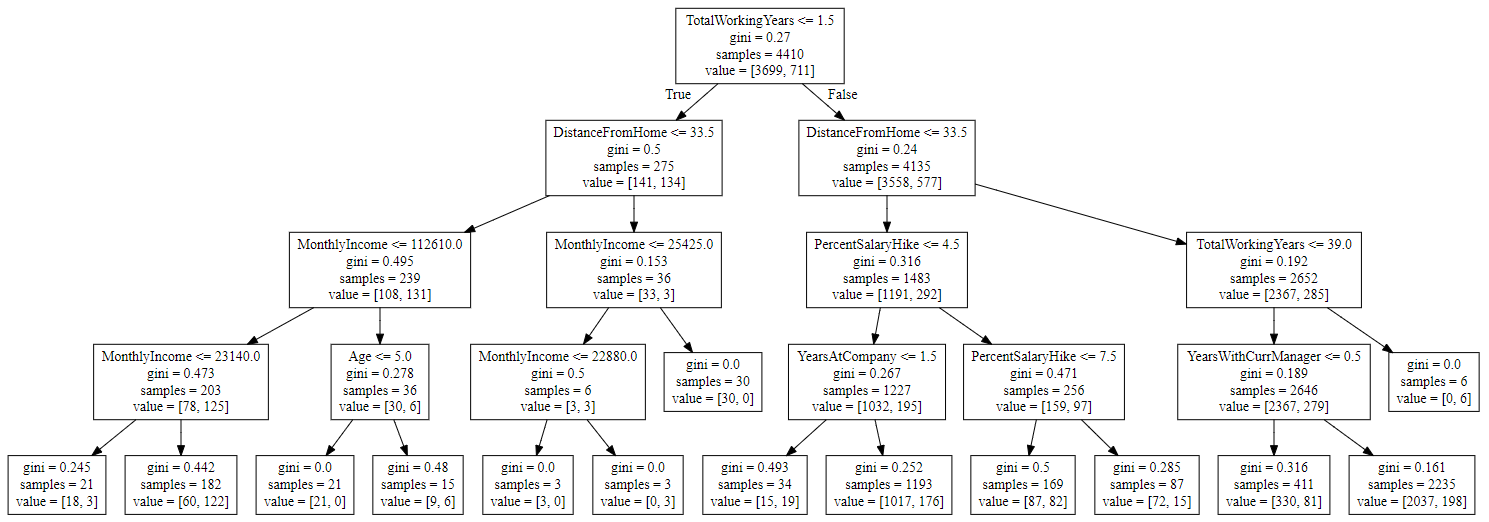
tree\_model=tree.DecisionTreeClassifier(max\_depth=4)

tree\_model.fit(X=predictors,y=attrition\_dataset["Attrition"])

with open("Attrition\_DTree.dot",'w') as f:

f=tree.export\_graphviz(tree\_model,feature\_names=["Age","DistanceFromHome","JobRole","MonthlyIncome","NumCompaniesWorked","PercentSalaryHike","TotalWorkingYears","YearsAtCompany","YearsWithCurrManager"],out\_file=f);

**Output:**



**Project 3: Build Decision Tree for Bank Loan Modelling**

**DV - "Personal Loan"**

**IDV - Output of RF Algorithm**

**RF Algo:**

#import packages

from sklearn.ensemble import RandomForestClassifier

import pandas as pd

import numpy as np

from sklearn import preprocessing

#load dataset

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

dataset=dataset.drop("ID",axis=1)

dataset=dataset.drop("ZIP Code",axis=1)

#checking for null values

print(dataset.isna().sum())

#build model

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=dataset[features],y=dataset["Personal Loan"])

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

#find important variable

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

print(features,imp)

**Output:**

OOB Accuracy score: 0.9882

Age 0.04502791859485629

Experience 0.043743840541234924

Income 0.3434413174028235

Family 0.10073132925290217

CCAvg 0.1797884656723131

Education 0.1644033879406872

Mortgage 0.0435998825383555

Securities Account 0.005750826829536648

CD Account 0.054856398467471555

Online 0.00835431455436651

CreditCard 0.010302318205452589

* **IDV:** Income, Family, CCAvg, Education are important variable considered for Decition Tree Algorithm

**Decision Tree Algo:**

#import packages

import pandas as pd

import numpy as np

from sklearn import tree

from sklearn import preprocessing

#load dataset dataset

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

#dropping unimportant variables

bank\_loan\_dataset=bank\_loan\_dataset.drop("ID",axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop("ZIP Code",axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('Age',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('Experience',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('Mortgage',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('Securities Account',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('CD Account',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('Online',axis=1)

bank\_loan\_dataset=bank\_loan\_dataset.drop('CreditCard',axis=1)

#checking for null values

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

#make the model

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

tree\_model=tree.DecisionTreeClassifier(max\_depth=5)

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

with open("Bank\_Loan\_DTree.dot",'w') as f:

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

**Output:**

