**Attrition Rate Analysis Decision Tree**

import pandas as pd

import numpy as np

from sklearn import preprocessing

from sklearn import tree

**Loading Data and Data Treatment:**

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

Attrition\_dataset.head(2)

Out[28]:

Age Attrition ... YearsSinceLastPromotion YearsWithCurrManager

0 51 No ... 0 0

1 31 Yes ... 1 4

[2 rows x 24 columns]

Attrition\_dataset.isnull().sum()

Out[29]:

Age 0

Attrition 0

BusinessTravel 0

Department 0

DistanceFromHome 0

Education 0

EducationField 0

EmployeeCount 0

EmployeeID 0

Gender 0

JobLevel 0

JobRole 0

MaritalStatus 0

MonthlyIncome 0

NumCompaniesWorked 19

Over18 0

PercentSalaryHike 0

StandardHours 0

StockOptionLevel 0

TotalWorkingYears 9

TrainingTimesLastYear 0

YearsAtCompany 0

YearsSinceLastPromotion 0

YearsWithCurrManager 0

dtype: int64

Attrition\_dataset.dtypes

Out[30]:

Age int64

Attrition object

BusinessTravel object

Department object

DistanceFromHome int64

Education int64

EducationField object

EmployeeCount int64

EmployeeID int64

Gender object

JobLevel int64

JobRole object

MaritalStatus object

MonthlyIncome int64

NumCompaniesWorked float64

Over18 object

PercentSalaryHike int64

StandardHours int64

StockOptionLevel int64

TotalWorkingYears float64

TrainingTimesLastYear int64

YearsAtCompany int64

YearsSinceLastPromotion int64

YearsWithCurrManager int64

dtype: object

Attrition\_dataset['NumCompaniesWorked'].mean()

Out[31]: 2.6948303347756775

Attrition\_dataset['TotalWorkingYears'].mean()

Out[32]: 11.279936378095888

Attrition\_dataset = Attrition\_dataset.fillna(Attrition\_dataset.mean().round())

**Encoding Categorical Variables**

label\_encoder = preprocessing.LabelEncoder()

Attrition\_dataset['Attrition']= label\_encoder.fit\_transform(Attrition\_dataset['Attrition'])

Attrition\_dataset['BusinessTravel']= label\_encoder.fit\_transform(Attrition\_dataset['BusinessTravel'])

Attrition\_dataset['Department']= label\_encoder.fit\_transform(Attrition\_dataset['Department'])

Attrition\_dataset['EducationField']= label\_encoder.fit\_transform(Attrition\_dataset['EducationField'])

Attrition\_dataset['Gender']= label\_encoder.fit\_transform(Attrition\_dataset['Gender'])

Attrition\_dataset['JobRole']= label\_encoder.fit\_transform(Attrition\_dataset['JobRole'])

Attrition\_dataset['MaritalStatus']= label\_encoder.fit\_transform(Attrition\_dataset['MaritalStatus'])

**Random Forest Algorithm to find imp Variables**

Attrition\_dataset.columns

Out[53]:

Index(['Age', 'Attrition', 'BusinessTravel', 'Department', 'DistanceFromHome',

'Education', 'EducationField', 'EmployeeCount', 'EmployeeID', 'Gender',

'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome',

'NumCompaniesWorked', 'Over18', 'PercentSalaryHike', 'StandardHours',

'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',

'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager'],

dtype='object')

features = ['Age', 'BusinessTravel', 'Department', 'DistanceFromHome',

'Education', 'EducationField', 'Gender',

'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome',

'NumCompaniesWorked', 'PercentSalaryHike',

'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',

'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager']

from sklearn.ensemble import RandomForestClassifier

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

rf\_model.fit(X= Attrition\_dataset[features], y= Attrition\_dataset['Attrition'])

Out[57]:

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: 1.0***

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

print(feature,imp)

Age 0.09702916713419754

BusinessTravel 0.028383175530811103

Department 0.02594778942540104

DistanceFromHome 0.0699537185410984

Education 0.04114250396800288

EducationField 0.04093127319538544

Gender 0.01875959921256577

JobLevel 0.03775469477514371

JobRole 0.05576471744208975

MaritalStatus 0.03957036297807382

MonthlyIncome 0.0942158017621201

NumCompaniesWorked 0.05594751802014477

PercentSalaryHike 0.0651308855684338

StockOptionLevel 0.034040045428423114

TotalWorkingYears 0.08463252503226504

TrainingTimesLastYear 0.04473475267255931

YearsAtCompany 0.06923680257829705

YearsSinceLastPromotion 0.0432388041164298

YearsWithCurrManager 0.05358586261855765

**Generating Decision Tree Model**

predictors = ['Age', 'MonthlyIncome', 'TotalWorkingYears']

tree\_model = tree.DecisionTreeClassifier(max\_depth= 6, max\_leaf\_nodes= 10)

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

Out[49]:

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

max\_depth=6, 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')

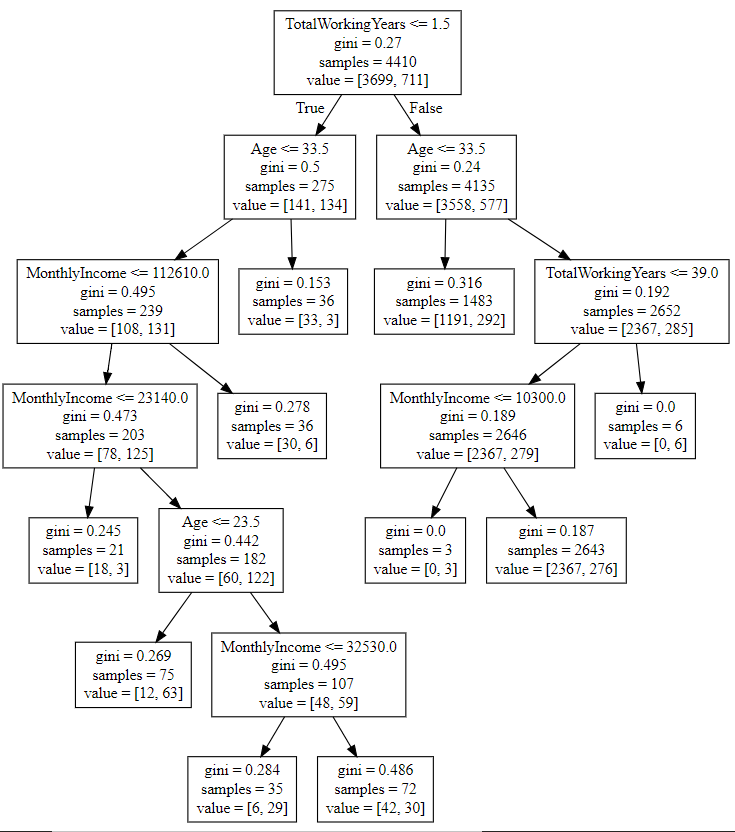
with open("Attrition\_DTree1.dot","w") as f:

f = tree.export\_graphviz(tree\_model,feature\_names= ['Age', 'MonthlyIncome', 'TotalWorkingYears'] , out\_file=f)

print("DTree Model Accuracy:", tree\_model.score(X= Attrition\_dataset[predictors], y= Attrition\_dataset['Attrition']))

***DTree Model Accuracy: 0.8575963718820862***

**Decision Tree:**



**Rules:**

**Attrition- NO**

1. If Total Working years is less than 1.5 and age is greater than 33.5, then there is low probability of Attrition
2. If Total Working years is less than 1.5, age is less than 33.5 and Monthly Income greater than 112610, then there is low probability of Attrition
3. If Total Working years is less than 1.5, age is less than 33.5 and Monthly Income less than 23140, then there is low probability of Attrition
4. If Total Working years is less than 1.5, age is in range 23.5 to 33.5 and Monthly Income is in range 32530 to 112610, then there is low probability of Attrition
5. If Total Working years is greater than 1.5 and age is less than 33.5, then there is low probability of Attrition
6. If Total Working years is in range of 1.5 to 39, age is greater than 33.5 and Monthly income greater than 10300, then there is low probability of Attrition

**Attrition- YES**

1. If Total Working years is greater than 39 and age is greater than 33.5, then there is high probability of Attrition
2. If Total Working years is in range of 1.5 to 39, age is greater than 33.5 and Monthly income less than 10300, then there is high probability of Attrition
3. If Total Working years is less than 1.5, age is less than 23.5 and Monthly Income is in range 23140 to 112160, then there is high probability of Attrition
4. If Total Working years is less than 1.5, age is in range 23.5 to 33.5 and Monthly Income is in range 23140 to 32530, then there is high probability of Attrition

**Inference:**

1. Based on the importance value generated with Random forest algorithm, it is seen that the features **‘Age', 'MonthlyIncome', and 'TotalWorkingYears'** are more significant for decision tree generation.
2. Increasing the no. of significant features and max-depth, increases the accuracy of the model. But the Decision tree becomes complex and overfitted.
3. Decision tree generated with these features and max-depth of 6 and 10 leaf nodes provides **85.76%** accuracy in classifying the record as Attrition(Y/N)