* Here,I find major causes of attrition in employees. Attrition results in lack of productivity which reults in lack of profit ,which ultimately leads to unemployment. It seems simple to point out few factors behind attrition such as low wages, working environment,relationship with boss etc, but actually there are a lot more from one's personal reason to his education. And this is where Statistics come in. This field of concern is so wide thatit has given birth to new ML research called "Interpretability".

# Importing the Packages

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# 2.Importing the data

attrition = pd.read\_csv('HR\_Employee\_Attrition\_Data.csv')

## ->There are no null values in the data

# 3.There are Some insignificant variables in the data

attrition.drop(['EmployeeCount','EmployeeNumber','Over18','StandardHours'],axis=1,inplace = True)

# 4.Converting the Categorical Variables in to dummy values

# 5. Split the data

# Import the train\_test\_split method

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(attrition, y, train\_size=0.8, random\_state=0)

# 6. Fitting Random Forest

seed = 0 # We set our random seed to zero for reproducibility

# Random Forest parameters

rf\_params = {

'n\_jobs': -1,

'n\_estimators': 1000,

# 'warm\_start': True,

'max\_features': 0.3,

'max\_depth': 4,

'min\_samples\_leaf': 2,

'max\_features' : 'sqrt',

'random\_state' : seed,

'verbose': 0

}

rf = RandomForestClassifier(\*\*rf\_params)

rf.fit(X\_train,y\_train)

rf\_predictions = rf.predict(X\_test)

print("Accuracy score: {}".format(accuracy\_score(y\_test, rf\_predictions)))

print("="\*100)

print(classification\_report(y\_test, rf\_predictions))

Accuracy score: 0.8435374149659864

====================================================================================================

precision recall f1-score support

0 0.84 1.00 0.91 488

1 1.00 0.08 0.15 100

avg / total 0.87 0.84 0.78 588

# 7.Feature Importance

import pandas as pd

feature\_imp = pd.Series(rf.feature\_importances\_,index=attrition.columns).sort\_values(ascending=False)

feature\_imp

## Based on importance of the variables less importance got removed

# Split dataset into features and labels

X=attrition[['OverTime','MonthlyIncome','TotalWorkingYears','YearsAtCompany','Age','StockOptionLevel','JobLevel','Marital\_single','YearsWithCurrManager','EnvironmentSatisfaction','Business\_Travel\_Frequently','YearsInCurrentRole','DistanceFromHome','DailyRate','JobSatisfaction','MonthlyRate',

'Sales Representative','JobInvolvement','HourlyRate','NumCompaniesWorked','WorkLifeBalance','PercentSalaryHike']]

# Split dataset into training set and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size=0.80, random\_state=0)

from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier

clf=RandomForestClassifier(n\_estimators=1000)

#Train the model using the training sets y\_pred=clf.predict(X\_test)

clf.fit(X\_train,y\_train)

# prediction on test set

y\_pred=clf.predict(X\_test)

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))

Accuracy: 0.9659863945578231

# 8. Fitting Neural network model

from sklearn.neural\_network import MLPClassifier

mlp = MLPClassifier(activation='relu', alpha=1e-5, solver = 'sgd', hidden\_layer\_sizes=(100))

mlp.fit(X\_train,y\_train)

predictions = mlp.predict(X\_test)

from sklearn.metrics import classification\_report,confusion\_matrix,accuracy\_score

accuracy\_score(y\_test, predictions)

0.8231292517006803

print(confusion\_matrix(y\_test, predictions))

print(classification\_report(y\_test,predictions))

precision recall f1-score support

0 0.82 1.00 0.90 484

1 0.00 0.00 0.00 104

avg / total 0.68 0.82 0.74 588

# Random forest with Feature Importance performed well than compared to Neural network model