HARI PRASATH S

225229110

LAB9:EMPLOYEE HOPPING PREDICTION USING RANDOM FOREST

STEP1[UNDERSTAND DATA]

In [1]:
```
import pandas as pd
```

In [2]:
```
data=pd.read_csv('Employee_hopping.csv')
data
```

17	22	No	Non-Travel	1123	Research & Development	16	2
18	53	No	Travel_Rarely	1219	Sales	2	4
19	38	No	Travel_Rarely	371	Research & Development	2	3
20	24	No	Non-Travel	673	Research & Development	11	2
21	36	Yes	Travel_Rarely	1218	Sales	9	4
22	34	No	Travel_Rarely	419	Research & Development	7	4
23	21	No	Travel_Rarely	391	Research & Development	15	2
24	34	Yes	Travel_Rarely	699	Research & Development	6	1
					Research &		

In [3]:
```
data.head()
```

Out[3]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Educati
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life S
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life S
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life S
4	27	No	Travel_Rarely	591	Research & Development	2	1	

5 rows × 35 columns

In [4]:
```
data.shape
```

Out[4]:
```
(1470, 35)
```

In [5]:
```
data.columns
```

Out[5]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

In [8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
Age                         1470 non-null int64
Attrition                   1470 non-null object
BusinessTravel              1470 non-null object
DailyRate                   1470 non-null int64
Department                  1470 non-null object
DistanceFromHome            1470 non-null int64
Education                   1470 non-null int64
EducationField              1470 non-null object
EmployeeCount               1470 non-null int64
EmployeeNumber              1470 non-null int64
EnvironmentSatisfaction     1470 non-null int64
Gender                      1470 non-null object
HourlyRate                  1470 non-null int64
JobInvolvement              1470 non-null int64
JobLevel                    1470 non-null int64
JobRole                     1470 non-null object
JobSatisfaction             1470 non-null int64
MaritalStatus               1470 non-null object
MonthlyIncome               1470 non-null int64
MonthlyRate                 1470 non-null int64
NumCompaniesWorked          1470 non-null int64
Over18                      1470 non-null object
OverTime                    1470 non-null object
PercentSalaryHike           1470 non-null int64
PerformanceRating           1470 non-null int64
RelationshipSatisfaction    1470 non-null int64
StandardHours               1470 non-null int64
StockOptionLevel            1470 non-null int64
TotalWorkingYears           1470 non-null int64
TrainingTimesLastYear       1470 non-null int64
WorkLifeBalance             1470 non-null int64
YearsAtCompany              1470 non-null int64
YearsInCurrentRole          1470 non-null int64
YearsSinceLastPromotion     1470 non-null int64
YearsWithCurrManager        1470 non-null int64
dtypes: int64(26), object(9)
memory usage: 402.0+ KB
```

In [10]:
```
data.dtypes
```

Out[10]:
```
Age                         int64
Attrition                  object
BusinessTravel             object
DailyRate                   int64
Department                 object
DistanceFromHome            int64
Education                   int64
EducationField             object
EmployeeCount               int64
EmployeeNumber              int64
EnvironmentSatisfaction     int64
Gender                     object
HourlyRate                  int64
JobInvolvement              int64
JobLevel                    int64
JobRole                    object
JobSatisfaction             int64
MaritalStatus              object
MonthlyIncome               int64
MonthlyRate                 int64
NumCompaniesWorked          int64
Over18                     object
OverTime                   object
PercentSalaryHike           int64
PerformanceRating           int64
RelationshipSatisfaction    int64
StandardHours               int64
StockOptionLevel            int64
TotalWorkingYears           int64
TrainingTimesLastYear       int64
WorkLifeBalance             int64
YearsAtCompany              int64
YearsInCurrentRole          int64
YearsSinceLastPromotion     int64
YearsWithCurrManager        int64
dtype: object
```

In [11]:
```
data['Age'].value_counts
```

Out[11]:
```
<bound method IndexOpsMixin.value_counts of 0       41
1       49
2       37
3       33
4       27
5       32
6       59
7       30
8       38
9       36
10      35
11      29
12      31
13      34
14      28
15      29
16      32
17      22
18      53
19      38
20      24
21      36
22      34
23      21
24      34
25      53
26      32
27      42
28      44
29      46
        ..
1440    36
1441    56
1442    29
1443    42
1444    56
1445    41
1446    34
1447    36
1448    41
1449    32
1450    35
1451    38
1452    50
1453    36
1454    45
1455    40
1456    35
1457    40
1458    35
1459    29
1460    29
1461    50
1462    39
1463    31
```

```
1464    26
1465    36
1466    39
1467    27
1468    49
1469    34
Name: Age, Length: 1470, dtype: int64>
```

STEP2[EXTRACT X AND Y]

In [12]:
```
X=data.drop('Attrition',axis=1)
X
```

ield	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	...	RelationshipSatisfaction	StandardHours	Stock
ces	1	1	2	...	1	80	
ces	1	2	3	...	4	80	
ther	1	4	4	...	2	80	
ces	1	5	4	...	3	80	
ical	1	7	1	...	4	80	
ces	1	8	4	...	3	80	
ical	1	10	3	...	1	80	

In [14]:
```
Y=data['Attrition'].values
Y
```

Out[14]:
```
array(['Yes', 'No', 'Yes', ..., 'No', 'No', 'No'], dtype=object)
```

In [15]:
```
Y=data.Attrition
```

In [16]:
```
Y
```

Out[16]:
```
0       Yes
1        No
2       Yes
3        No
4        No
5        No
6        No
7        No
8        No
9        No
10       No
11       No
12       No
13       No
14      Yes
15       No
16       No
17       No
18       No
19       No
20       No
21      Yes
22       No
23       No
24      Yes
25       No
26      Yes
27       No
28       No
29       No
       ...
1440     No
1441     No
1442    Yes
1443     No
1444    Yes
1445     No
1446     No
1447     No
1448     No
1449     No
1450     No
1451     No
1452    Yes
1453     No
1454     No
1455     No
1456     No
1457     No
1458     No
1459     No
1460     No
1461    Yes
1462     No
1463     No
```

```
1464     No
1465     No
1466     No
1467     No
1468     No
1469     No
Name: Attrition, Length: 1470, dtype: object
```

STEP3:[FEATURE ENGINEERING]

In [25]:
```
data = pd.get_dummies(data, columns = ['BusinessTravel','Department','EducationFi
data
```

Out[25]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeNumber	EnvironmentSa
0	41	Yes	1102	1	2	1	
1	49	No	279	8	1	2	
2	37	Yes	1373	2	2	4	
3	33	No	1392	3	4	5	
4	27	No	591	2	1	7	
5	32	No	1005	2	2	8	
6	59	No	1324	3	3	10	
7	30	No	1358	24	1	11	
8	38	No	216	23	3	12	
9	36	No	1299	27	3	13	
10	35	No	809	16	3	14	
11	29	No	153	15	2	15	
12	31	No	670	26	1	16	
13	34	No	1346	19	2	18	
14	28	Yes	103	24	3	19	
15	29	No	1389	21	4	20	
16	32	No	334	5	2	21	
17	22	No	1123	16	2	22	
18	53	No	1219	2	4	23	
19	38	No	371	2	3	24	
20	24	No	673	11	2	26	
21	36	Yes	1218	9	4	27	
22	34	No	419	7	4	28	
23	21	No	391	15	2	30	
24	34	Yes	699	6	1	31	
25	53	No	1282	5	3	32	
26	32	Yes	1125	16	1	33	
27	42	No	691	8	4	35	
28	44	No	477	7	4	36	
29	46	No	705	2	4	38	
...	...	...	...	...	...	...	
1440	36	No	688	4	2	2025	

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeNumber	EnvironmentSa
1441	56	No	667	1	4	2026	
1442	29	Yes	1092	1	4	2027	
1443	42	No	300	2	3	2031	
1444	56	Yes	310	7	2	2032	
1445	41	No	582	28	4	2034	
1446	34	No	704	28	3	2035	
1447	36	No	301	15	4	2036	
1448	41	No	930	3	3	2037	
1449	32	No	529	2	3	2038	
1450	35	No	1146	26	4	2040	
1451	38	No	345	10	2	2041	
1452	50	Yes	878	1	4	2044	
1453	36	No	1120	11	4	2045	
1454	45	No	374	20	3	2046	
1455	40	No	1322	2	4	2048	
1456	35	No	1199	18	4	2049	
1457	40	No	1194	2	4	2051	
1458	35	No	287	1	4	2052	
1459	29	No	1378	13	2	2053	
1460	29	No	468	28	4	2054	
1461	50	Yes	410	28	3	2055	
1462	39	No	722	24	1	2056	
1463	31	No	325	5	3	2057	
1464	26	No	1167	5	3	2060	
1465	36	No	884	23	2	2061	
1466	39	No	613	6	1	2062	
1467	27	No	155	4	3	2064	
1468	49	No	1023	2	3	2065	
1469	34	No	628	8	3	2068	

1470 rows × 56 columns

STEP4:SHAPE OF X AND Y

In [26]:
```
X = data.drop(['Attrition'],axis=1)
print('X Shape : ',X.shape)
print('y Shape : ',Y.shape)
```

```
X Shape :  (1470, 55)
y Shape :  (1470,)
```

STEP5:[MODEL DEVELOPMENT]

In [33]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [28]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size =0.2, random_s
```

In [30]:
```
X_train.shape
```

Out[30]:
```
(1176, 55)
```

In [31]:
```
Y_train.shape
```

Out[31]:
```
(1176,)
```

In [29]:
```
from sklearn.ensemble import RandomForestClassifier
RFC = RandomForestClassifier(n_estimators=100, max_features=0.3)
```

In [32]:
```
RFC.fit(X_train,Y_train)
```

Out[32]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features=0.3, 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, n_estimators=100, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [34]:
```
RFC_y_pred = RFC.predict(X_test)
RFC_y_pred
```

Out[34]:
```
array(['No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No'], dtype=object)
```

STEP6: TESTING

In [36]:
```
from sklearn.metrics import accuracy_score,classification_report
```

In [37]:
```
RFC_acc=accuracy_score(Y_test,RFC_y_pred)
RFC_acc
```

Out[37]:
```
0.8741496598639455
```

In [38]:
```
print(classification_report(Y_test, RFC_y_pred))
```

```
             precision    recall  f1-score   support

         No       0.88      0.99      0.93       255
        Yes       0.62      0.13      0.21        39

avg / total       0.85      0.87      0.84       294
```

STEP7:[FEATURE IMPORTANCE VALUE]

In [40]:
```
print(RFC.feature_importances_)
```

```
[0.05607425 0.04847613 0.0423185  0.01840366 0.04512837 0.02352631
 0.03830164 0.02089741 0.0220411  0.02328977 0.08924022 0.03706093
 0.03302046 0.02661394 0.002665   0.01815755 0.         0.02776113
 0.04939884 0.02292581 0.01825114 0.03679423 0.02665785 0.02373052
 0.02550681 0.00297979 0.01147737 0.00572122 0.00234421 0.00744924
 0.00824727 0.00223425 0.00425583 0.00439705 0.00564054 0.00263834
 0.00811429 0.         0.00511024 0.00458517 0.0015612  0.00327906
 0.00614142 0.00105966 0.00184353 0.00057371 0.00485413 0.00621485
 0.00843136 0.00537385 0.00525984 0.02129422 0.         0.04583557
 0.03684123]
```

In [42]:
```
feature_name = pd.DataFrame(RFC.feature_importances_, index=X_train.columns, colu
feature_name
```

Out[42]:

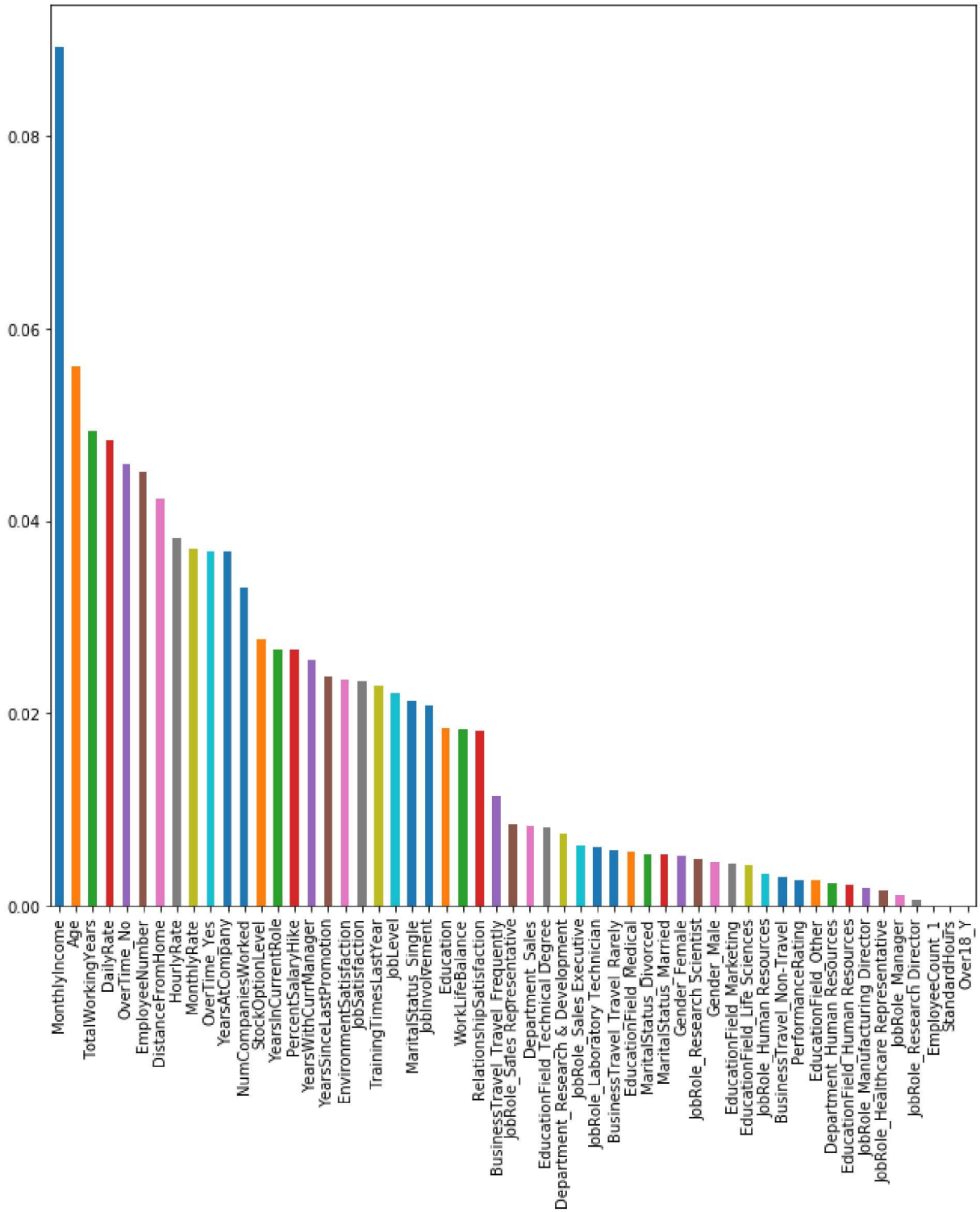
	Important_Feature
Age	0.056074
DailyRate	0.048476
DistanceFromHome	0.042318
Education	0.018404
EmployeeNumber	0.045128
EnvironmentSatisfaction	0.023526
HourlyRate	0.038302
JobInvolvement	0.020897
JobLevel	0.022041
JobSatisfaction	0.023290
MonthlyIncome	0.089240
MonthlyRate	0.037061
NumCompaniesWorked	0.033020
PercentSalaryHike	0.026614
PerformanceRating	0.002665
RelationshipSatisfaction	0.018158
StandardHours	0.000000
StockOptionLevel	0.027761
TotalWorkingYears	0.049399
TrainingTimesLastYear	0.022926
WorkLifeBalance	0.018251
YearsAtCompany	0.036794
YearsInCurrentRole	0.026658
YearsSinceLastPromotion	0.023731
YearsWithCurrManager	0.025507
BusinessTravel_Non-Travel	0.002980
BusinessTravel_Travel_Frequently	0.011477
BusinessTravel_Travel_Rarely	0.005721
Department_Human Resources	0.002344
Department_Research & Development	0.007449
Department_Sales	0.008247
EducationField_Human Resources	0.002234
EducationField_Life Sciences	0.004256

	Important_Feature
EducationField_Marketing	0.004397
EducationField_Medical	0.005641
EducationField_Other	0.002638
EducationField_Technical Degree	0.008114
EmployeeCount_1	0.000000
Gender_Female	0.005110
Gender_Male	0.004585
JobRole_Healthcare Representative	0.001561
JobRole_Human Resources	0.003279
JobRole_Laboratory Technician	0.006141
JobRole_Manager	0.001060
JobRole_Manufacturing Director	0.001844
JobRole_Research Director	0.000574
JobRole_Research Scientist	0.004854
JobRole_Sales Executive	0.006215
JobRole_Sales Representative	0.008431
MaritalStatus_Divorced	0.005374
MaritalStatus_Married	0.005260
MaritalStatus_Single	0.021294
Over18_Y	0.000000
OverTime_No	0.045836
OverTime_Yes	0.036841

In [43]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [46]:
```
pd.Series(RFC.feature_importances_, index=X_train.columns).sort_values(ascending=
```

Out[46]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1dcbd0bada0>
```



STEP8: Visualize your RF Decision Tree using graphviz

In [71]:
```
estimator = RFC.estimators_[5]
```

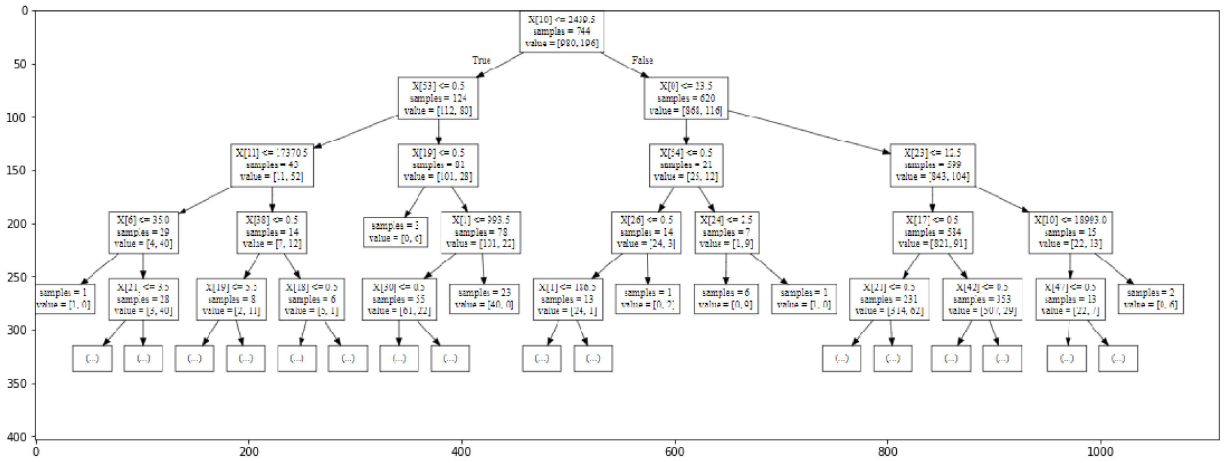
In [75]:
```
from sklearn import tree
from sklearn.tree import export_graphviz
with open("RFDT.dot", 'w') as f:
    f = tree.export_graphviz(estimator, out_file=f, max_depth=4, impurity=False)
```

In [76]:
```
!dot - Tpng RFDT.dot -o RFDT.png
```

```
'dot' is not recognized as an internal or external command,
operable program or batch file.
```

In [77]:
```
import matplotlib.pyplot as plt
image = plt.imread('RFDT.png')
plt.figure(figsize=(19,15))
plt.imshow(image)
```

Out[77]: <matplotlib.image.AxesImage at 0x1dcbe26df98>



STEP9:RF WITH A RANGE OF TREES

In [56]:
```
rf2 = RandomForestClassifier(oob_score=True, random_state=42, warm_start=True, n_
oob_list = list()
for n_trees in [15, 20, 30, 40, 50, 100, 150, 200, 300, 400]:
    rf2.set_params(n_estimators=n_trees)
    rf2.fit(X_train, Y_train)
    oob_error = 1 - rf2.oob_score_
    oob_list.append(pd.Series({'n_trees': n_trees, 'oob': oob_error}))
rf_oob_df = pd.concat(oob_list, axis=1).T.set_index('n_trees')
rf_oob_df
```

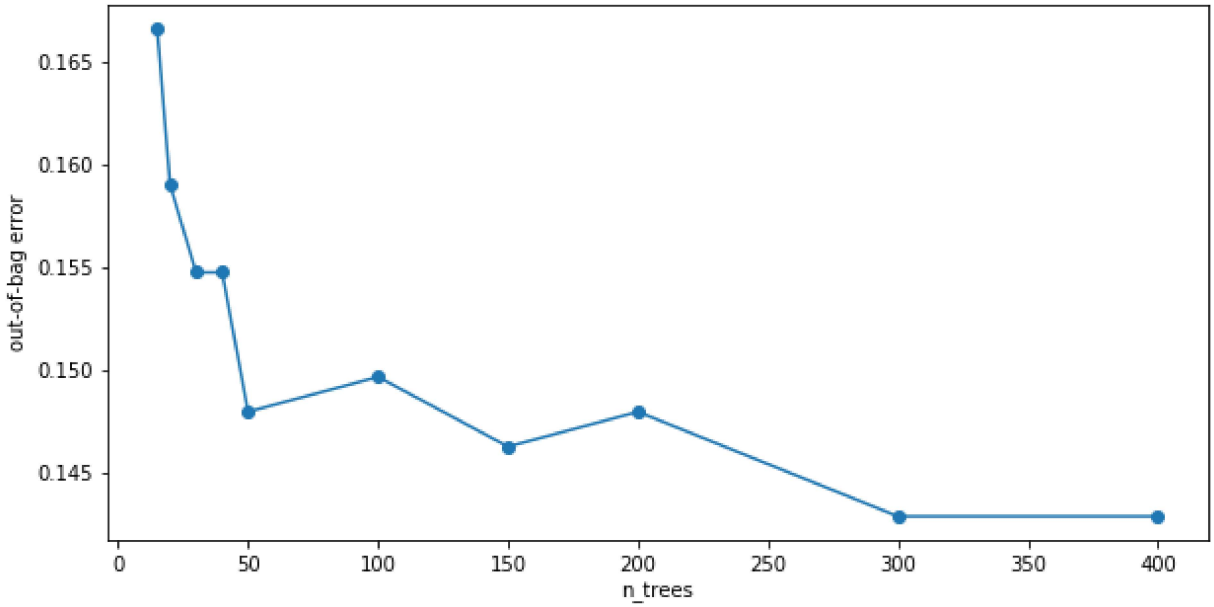
Out[56]:

n_trees	oob
15.0	0.166667
20.0	0.159014
30.0	0.154762
40.0	0.154762
50.0	0.147959
100.0	0.149660
150.0	0.146259
200.0	0.147959
300.0	0.142857
400.0	0.142857

STEP10: PLOT OOB -ERROR FOR EACH TREE

In [57]:
```
ax = rf_oob_df.plot(legend=False, marker='o', figsize=(10,5))
ax.set(ylabel='out-of-bag error')
```

Out[57]:
```
[Text(0,0.5,'out-of-bag error')]
```



STEP11: COMPARE WITH DECISION TREE CLASSIFIER

In [59]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,classification_report
clf = DecisionTreeClassifier(max_depth=4, random_state=42)
clf.fit(X_test,Y_test)
```

Out[59]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=4,
            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=False, random_state=42,
            splitter='best')
```

In [60]:
```python
y_pred1 = clf.predict(X_test)
y_pred1
```

Out[60]:
```
array(['No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'Yes', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No'], dtype=object)
```

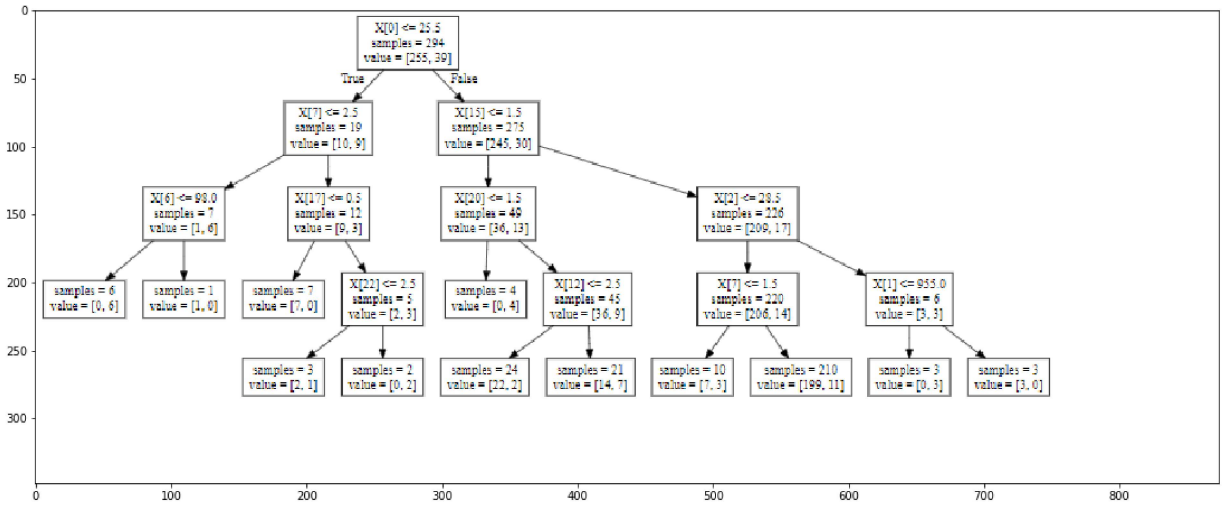
In [79]:
```python
from sklearn import tree
from sklearn.tree import export_graphviz
with open("DTC2.dot", 'w') as f:
    f = tree.export_graphviz(clf,out_file=f,max_depth = 4,impurity = False)
```

In [80]:
```python
!dot -Tpng DTC2.dot -o DTC2.png
```

```
'dot' is not recognized as an internal or external command,
operable program or batch file.
```

In [81]:
```
image = plt.imread('DTC2.png')
plt.figure(figsize=(19,15))
plt.imshow(image)
```

Out[81]: <matplotlib.image.AxesImage at 0x1dcbe1d0c50>



In [82]:
```
print("Accuracy of test :",clf.score(X_test,Y_test))
```

```
Accuracy of test : 0.9183673469387755
```

In [83]:
```
print(classification_report(Y_test,RFC_y_pred))
```

```
             precision    recall  f1-score   support

         No       0.88      0.99      0.93       255
        Yes       0.62      0.13      0.21        39

avg / total       0.85      0.87      0.84       294
```

In [ ]: