Lab9 - Employee Hopping Prediction using Random Forests

Name : Ezhilarasan C

Roll No : 225229151

Step1

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

In [2]:
```
df=pd.read_csv("Employee_Hopping.csv")
df.head()
```

Out[2]:

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

5 rows × 35 columns

In [3]:
```
df.shape
```

Out[3]: (1470, 35)

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCoun
t',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRat
e',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWor
ked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotio
n',
       'YearsWithCurrManager'],
      dtype='object')
```

In [6]:

```
df.dtypes
```

Out[6]:

```
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 [7]:
```
df.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 [9]:
```
df.Attrition.value_counts()
```

Out[9]:
```
No     1233
Yes     237
Name: Attrition, dtype: int64
```

Step2

In [10]:
```
X = df.drop(['Attrition'], axis=1)
y = df.Attrition
```

Step3

In [11]:
```
X_cat_cols = X.select_dtypes('object')
X_cat_cols
```

Out[11]:

	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus
0	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Single
1	Travel_Frequently	Research & Development	Life Sciences	Male	Research Scientist	Married
2	Travel_Rarely	Research & Development	Other	Male	Laboratory Technician	Single
3	Travel_Frequently	Research & Development	Life Sciences	Female	Research Scientist	Married
4	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married
5	Travel_Frequently	Research & Development	Life Sciences	Male	Laboratory Technician	Single
6	Travel_Rarely	Research & Development	Medical	Female	Laboratory Technician	Married
7	Travel_Rarely	Research & Development	Life Sciences	Male	Laboratory Technician	Divorced
8	Travel_Frequently	Research & Development	Life Sciences	Male	Manufacturing Director	Single
9	Travel_Rarely	Research & Development	Medical	Male	Healthcare Representative	Married
10	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married
11	Travel_Rarely	Research & Development	Life Sciences	Female	Laboratory Technician	Single
12	Travel_Rarely	Research & Development	Life Sciences	Male	Research Scientist	Divorced
13	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Divorced
14	Travel_Rarely	Research & Development	Life Sciences	Male	Laboratory Technician	Single
15	Travel_Rarely	Research & Development	Life Sciences	Female	Manufacturing Director	Divorced
16	Travel_Rarely	Research & Development	Life Sciences	Male	Research Scientist	Divorced
17	Non-Travel	Research & Development	Medical	Male	Laboratory Technician	Divorced
18	Travel_Rarely	Sales	Life Sciences	Female	Manager	Married
19	Travel_Rarely	Research & Development	Life Sciences	Male	Research Scientist	Single
20	Non-Travel	Research & Development	Other	Female	Manufacturing Director	Divorced
21	Travel_Rarely	Sales	Life Sciences	Male	Sales Representative	Single
22	Travel_Rarely	Research & Development	Life Sciences	Female	Research Director	Single
23	Travel_Rarely	Research & Development	Life Sciences	Male	Research Scientist	Single

	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus
24	Travel_Rarely	Research & Development	Medical	Male	Research Scientist	Single
25	Travel_Rarely	Research & Development	Other	Female	Manager	Divorced
26	Travel_Frequently	Research & Development	Life Sciences	Female	Research Scientist	Single
27	Travel_Rarely	Sales	Marketing	Male	Sales Executive	Married
28	Travel_Rarely	Research & Development	Medical	Female	Healthcare Representative	Married
29	Travel_Rarely	Sales	Marketing	Female	Manager	Single
...	...	...	...	...	...	...
1440	Travel_Frequently	Research & Development	Life Sciences	Female	Manufacturing Director	Divorced
1441	Non-Travel	Research & Development	Life Sciences	Male	Healthcare Representative	Divorced
1442	Travel_Rarely	Research & Development	Medical	Male	Research Scientist	Married
1443	Travel_Rarely	Research & Development	Life Sciences	Male	Manager	Married
1444	Travel_Rarely	Research & Development	Technical Degree	Male	Laboratory Technician	Married
1445	Travel_Rarely	Research & Development	Life Sciences	Female	Manufacturing Director	Married
1446	Travel_Rarely	Sales	Marketing	Female	Sales Executive	Married
1447	Non-Travel	Sales	Marketing	Male	Sales Executive	Divorced
1448	Travel_Rarely	Sales	Life Sciences	Male	Sales Executive	Divorced
1449	Travel_Rarely	Research & Development	Technical Degree	Male	Research Scientist	Single
1450	Travel_Rarely	Human Resources	Life Sciences	Female	Human Resources	Single
1451	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Married
1452	Travel_Frequently	Sales	Life Sciences	Male	Sales Executive	Divorced
1453	Travel_Rarely	Sales	Marketing	Female	Sales Executive	Married
1454	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Single
1455	Travel_Rarely	Research & Development	Life Sciences	Male	Research Scientist	Single
1456	Travel_Frequently	Research & Development	Life Sciences	Male	Healthcare Representative	Married

	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus
1457	Travel_Rarely	Research & Development	Medical	Female	Research Scientist	Married
1458	Travel_Rarely	Research & Development	Life Sciences	Female	Research Scientist	Married
1459	Travel_Rarely	Research & Development	Other	Male	Laboratory Technician	Married
1460	Travel_Rarely	Research & Development	Medical	Female	Research Scientist	Single
1461	Travel_Rarely	Sales	Marketing	Male	Sales Executive	Divorced
1462	Travel_Rarely	Sales	Marketing	Female	Sales Executive	Married
1463	Non-Travel	Research & Development	Medical	Male	Manufacturing Director	Single
1464	Travel_Rarely	Sales	Other	Female	Sales Representative	Single
1465	Travel_Frequently	Research & Development	Medical	Male	Laboratory Technician	Married
1466	Travel_Rarely	Research & Development	Medical	Male	Healthcare Representative	Married
1467	Travel_Rarely	Research & Development	Life Sciences	Male	Manufacturing Director	Married
1468	Travel_Frequently	Sales	Medical	Male	Sales Executive	Married
1469	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married

1470 rows × 8 columns

In [12]:

```
df.select_dtypes(include=['object']).dtypes
```

Out[12]:

```
Attrition         object
BusinessTravel    object
Department        object
EducationField    object
Gender            object
JobRole           object
MaritalStatus     object
Over18            object
OverTime          object
dtype: object
```

In [13]:
```
df = pd.get_dummies(df, columns=["BusinessTravel", "Department","Educatior
                                  "Over18","OverTime"])
df.head()
```

Out[13]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNum
0	41	Yes	1102	1	2	1	
1	49	No	279	8	1	1	
2	37	Yes	1373	2	2	1	
3	33	No	1392	3	4	1	
4	27	No	591	2	1	1	

5 rows × 56 columns

Step4

In [14]:
```
X = df.drop(['Attrition'], axis=1)
X.shape
```

Out[14]: (1470, 55)

In [15]:
```
y.shape
```

Out[15]: (1470,)

Step5

In [18]:
```
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, r
```

In [19]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=1000, max_features=10)
rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)
```

Step6

In [21]:
```
from sklearn.metrics import accuracy_score, classification_report
print("Accuracy =", accuracy_score(y_test, y_pred),'\n')
print("Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy = 0.8707482993197279

Report:
              precision    recall  f1-score   support

         No       0.88      0.99      0.93       255
        Yes       0.57      0.10      0.17        39

avg / total       0.84      0.87      0.83       294
```

Step7

In [22]:
```
print(rfc.feature_importances_)
```

```
[0.0517451  0.04722933 0.03887903 0.01710468 0.         0.04278778
 0.02311509 0.03904183 0.02021352 0.02396834 0.02326452 0.07920932
 0.04228448 0.03307936 0.02732406 0.00401553 0.01840954 0.
 0.02848345 0.04758049 0.02328496 0.01992852 0.03812122 0.02614615
 0.0234592  0.02751905 0.0034132  0.01215741 0.00658376 0.00183751
 0.00747679 0.00872864 0.00227644 0.00603419 0.00537846 0.00632019
 0.00289069 0.00698804 0.0054528  0.00646902 0.00178921 0.00254453
 0.00749878 0.00110771 0.00242763 0.00072916 0.0065639  0.00711959
 0.00828513 0.00509135 0.00661428 0.0192785  0.         0.04180135
 0.04094719]
```

In [23]:
```
feature_name = pd.DataFrame(rfc.feature_importances_, index=X_train.columr
feature_name
```

Out[23]:

	Important_Feature
Age	0.051745
DailyRate	0.047229
DistanceFromHome	0.038879
Education	0.017105
EmployeeCount	0.000000
EmployeeNumber	0.042788
EnvironmentSatisfaction	0.023115
HourlyRate	0.039042
JobInvolvement	0.020214
JobLevel	0.023968
JobSatisfaction	0.023265
MonthlyIncome	0.079209
MonthlyRate	0.042284
NumCompaniesWorked	0.033079
PercentSalaryHike	0.027324
PerformanceRating	0.004016
RelationshipSatisfaction	0.018410
StandardHours	0.000000
StockOptionLevel	0.028483
TotalWorkingYears	0.047580
TrainingTimesLastYear	0.023285
WorkLifeBalance	0.019929
YearsAtCompany	0.038121
YearsInCurrentRole	0.026146
YearsSinceLastPromotion	0.023459
YearsWithCurrManager	0.027519
BusinessTravel_Non-Travel	0.003413
BusinessTravel_Travel_Frequently	0.012157
BusinessTravel_Travel_Rarely	0.006584
Department_Human Resources	0.001838
Department_Research & Development	0.007477
Department_Sales	0.008729
EducationField_Human Resources	0.002276
EducationField_Life Sciences	0.006034
EducationField_Marketing	0.005378
EducationField_Medical	0.006320

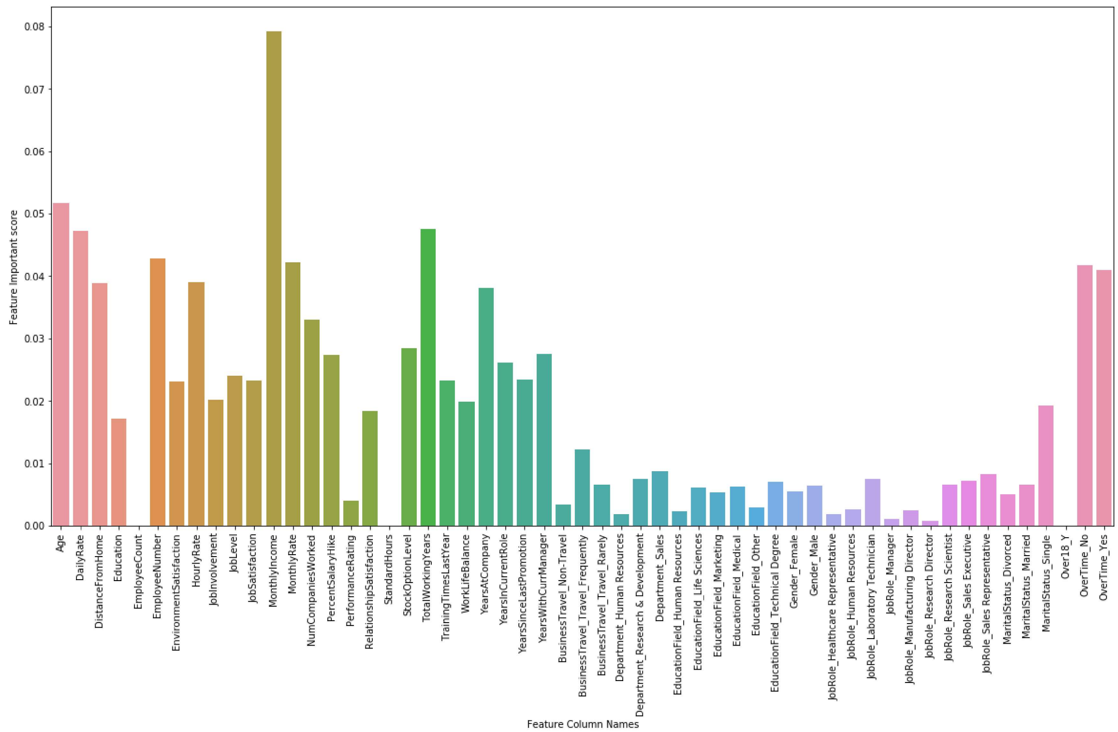
	Important_Feature
EducationField_Other	0.002891
EducationField_Technical Degree	0.006988
Gender_Female	0.005453
Gender_Male	0.006469
JobRole_Healthcare Representative	0.001789
JobRole_Human Resources	0.002545
JobRole_Laboratory Technician	0.007499
JobRole_Manager	0.001108
JobRole_Manufacturing Director	0.002428
JobRole_Research Director	0.000729
JobRole_Research Scientist	0.006564
JobRole_Sales Executive	0.007120
JobRole_Sales Representative	0.008285
MaritalStatus_Divorced	0.005091
MaritalStatus_Married	0.006614
MaritalStatus_Single	0.019279
Over18_Y	0.000000
OverTime_No	0.041801
OverTime_Yes	0.040947

In [24]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [30]:
```
plt.figure(figsize=(20,10))
sns.barplot(x=feature_name.index, y=feature_name['Important_Feature'])
plt.xticks(rotation=90)
plt.xlabel('Feature Column Names')
plt.ylabel('Feature Important score')
```

Out[30]:
```
Text(0,0.5,'Feature Important score')
```



Step8

In [31]:
```
from sklearn import tree
```

In [33]:
```
from sklearn.tree import export_graphviz
```

In [34]:
```
estim = rfc.estimators_[5]
estim
```

Out[34]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=No
ne,
            max_features=10, 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=1460894565, splitter='best')
```

In [35]:

```
with open("tree.dot",'w') as f:
    f = tree.export_graphviz(estim, out_file = f, max_depth = 4, impurity=
        class_names=['Yes', 'No'], filled=True)
```

In [36]:

```
!type tree.dot
```

```
digraph Tree {
node [shape=box, style="filled", color="black"] ;
0 [label="TotalWorkingYears <= 1.5\nsamples = 749\nvalue = [954, 222]\nc
lass = Yes", fillcolor="#e58139c4"] ;
1 [label="OverTime_Yes <= 0.5\nsamples = 51\nvalue = [31, 47]\nclass = N
o", fillcolor="#399de557"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="DailyRate <= 355.0\nsamples = 33\nvalue = [26, 24]\nclass = Ye
s", fillcolor="#e5813914"] ;
1 -> 2 ;
3 [label="samples = 6\nvalue = [0, 8]\nclass = No", fillcolor="#399de5f
f"] ;
2 -> 3 ;
4 [label="WorkLifeBalance <= 2.5\nsamples = 27\nvalue = [26, 16]\nclass
= Yes", fillcolor="#e5813962"] ;
2 -> 4 ;
5 [label="PercentSalaryHike <= 16.5\nsamples = 7\nvalue = [3, 6]\nclass
= No", fillcolor="#399de57f"] ;
4 -> 5 ;
6 [label="(...)", fillcolor="#C0C0C0"] ;
5 -> 6 ;
11 [label="(...)", fillcolor="#C0C0C0"] ;
5 -> 11 ;
12 [label="EnvironmentSatisfaction <= 2.5\nsamples = 20\nvalue = [23, 1
0]\nclass = Yes", fillcolor="#e5813990"] ;
4 -> 12 ;
13 [label="(...)", fillcolor="#C0C0C0"] ;
12 -> 13 ;
16 [label="(...)", fillcolor="#C0C0C0"] ;
12 -> 16 ;
19 [label="StockOptionLevel <= 0.5\nsamples = 18\nvalue = [5, 23]\nclass
= No", fillcolor="#399de5c8"] ;
1 -> 19 ;
20 [label="samples = 11\nvalue = [0, 18]\nclass = No", fillcolor="#399de
5ff"] ;
19 -> 20 ;
21 [label="MaritalStatus_Divorced <= 0.5\nsamples = 7\nvalue = [5, 5]\nc
lass = Yes", fillcolor="#e5813900"] ;
19 -> 21 ;
22 [label="MonthlyIncome <= 2269.5\nsamples = 4\nvalue = [1, 4]\nclass =
No", fillcolor="#399de5bf"] ;
21 -> 22 ;
23 [label="(...)", fillcolor="#C0C0C0"] ;
22 -> 23 ;
24 [label="(...)", fillcolor="#C0C0C0"] ;
22 -> 24 ;
25 [label="MonthlyRate <= 25972.5\nsamples = 3\nvalue = [4, 1]\nclass =
Yes", fillcolor="#e58139bf"] ;
21 -> 25 ;
26 [label="(...)", fillcolor="#C0C0C0"] ;
25 -> 26 ;
27 [label="(...)", fillcolor="#C0C0C0"] ;
25 -> 27 ;
28 [label="OverTime_No <= 0.5\nsamples = 698\nvalue = [923, 175]\nclass
= Yes", fillcolor="#e58139cf"] ;
0 -> 28 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
29 [label="YearsAtCompany <= 3.5\nsamples = 204\nvalue = [235, 98]\nclas
```

```
s = Yes", fillcolor="#e5813995"] ;
28 -> 29 ;
30 [label="DistanceFromHome <= 1.5\nsamples = 59\nvalue = [49, 49]\nclas
s = Yes", fillcolor="#e5813900"] ;
29 -> 30 ;
31 [label="WorkLifeBalance <= 3.5\nsamples = 8\nvalue = [2, 12]\nclass =
No", fillcolor="#399de5d4"] ;
30 -> 31 ;
32 [label="(...)", fillcolor="#C0C0C0"] ;
31 -> 32 ;
37 [label="(...)", fillcolor="#C0C0C0"] ;
31 -> 37 ;
38 [label="PercentSalaryHike <= 12.5\nsamples = 51\nvalue = [47, 37]\ncl
ass = Yes", fillcolor="#e5813936"] ;
30 -> 38 ;
39 [label="(...)", fillcolor="#C0C0C0"] ;
38 -> 39 ;
44 [label="(...)", fillcolor="#C0C0C0"] ;
38 -> 44 ;
63 [label="YearsAtCompany <= 10.5\nsamples = 145\nvalue = [186, 49]\ncla
ss = Yes", fillcolor="#e58139bc"] ;
29 -> 63 ;
64 [label="YearsWithCurrManager <= 6.5\nsamples = 111\nvalue = [135, 44]
\nclass = Yes", fillcolor="#e58139ac"] ;
63 -> 64 ;
65 [label="(...)", fillcolor="#C0C0C0"] ;
64 -> 65 ;
100 [label="(...)", fillcolor="#C0C0C0"] ;
64 -> 100 ;
119 [label="MonthlyRate <= 2658.0\nsamples = 34\nvalue = [51, 5]\nclass
= Yes", fillcolor="#e58139e6"] ;
63 -> 119 ;
120 [label="(...)", fillcolor="#C0C0C0"] ;
119 -> 120 ;
125 [label="(...)", fillcolor="#C0C0C0"] ;
119 -> 125 ;
128 [label="EducationField_Technical Degree <= 0.5\nsamples = 494\nvalue
= [688, 77]\nclass = Yes", fillcolor="#e58139e2"] ;
28 -> 128 ;
129 [label="NumCompaniesWorked <= 5.5\nsamples = 447\nvalue = [634, 58]
\nclass = Yes", fillcolor="#e58139e8"] ;
128 -> 129 ;
130 [label="JobSatisfaction <= 1.5\nsamples = 373\nvalue = [546, 30]\ncl
ass = Yes", fillcolor="#e58139f1"] ;
129 -> 130 ;
131 [label="(...)", fillcolor="#C0C0C0"] ;
130 -> 131 ;
154 [label="(...)", fillcolor="#C0C0C0"] ;
130 -> 154 ;
195 [label="JobInvolvement <= 2.5\nsamples = 74\nvalue = [88, 28]\nclass
= Yes", fillcolor="#e58139ae"] ;
129 -> 195 ;
196 [label="(...)", fillcolor="#C0C0C0"] ;
195 -> 196 ;
209 [label="(...)", fillcolor="#C0C0C0"] ;
195 -> 209 ;
222 [label="DailyRate <= 1354.5\nsamples = 47\nvalue = [54, 19]\nclass =
```

```
Yes", fillcolor="#e58139a5"] ;
128 -> 222 ;
223 [label="HourlyRate <= 98.5\nsamples = 35\nvalue = [45, 10]\nclass =
Yes", fillcolor="#e58139c6"] ;
222 -> 223 ;
224 [label="(...)", fillcolor="#C0C0C0"] ;
223 -> 224 ;
239 [label="(...)", fillcolor="#C0C0C0"] ;
223 -> 239 ;
242 [label="EmployeeNumber <= 1028.0\nsamples = 12\nvalue = [9, 9]\nclas
s = Yes", fillcolor="#e5813900"] ;
222 -> 242 ;
243 [label="(...)", fillcolor="#C0C0C0"] ;
242 -> 243 ;
246 [label="(...)", fillcolor="#C0C0C0"] ;
242 -> 246 ;
}
```

Step9

In [38]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [39]:

```
rf2 = RandomForestClassifier(oob_score=True, random_state=42, warm_start=1
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
```

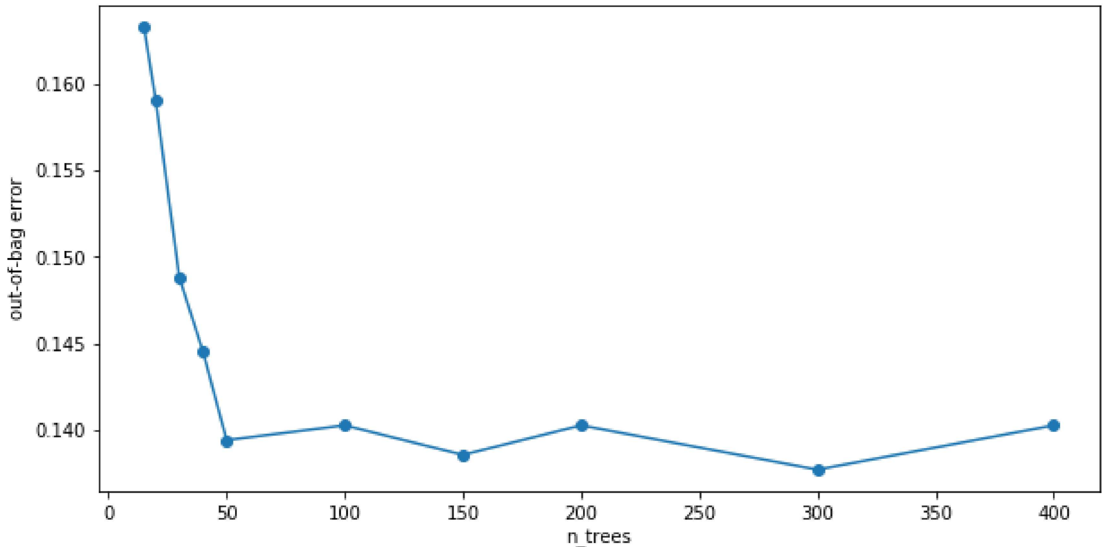
Out[39]:

n_trees	oob
15.0	0.163265
20.0	0.159014
30.0	0.148810
40.0	0.144558
50.0	0.139456
100.0	0.140306
150.0	0.138605
200.0	0.140306
300.0	0.137755
400.0	0.140306

Step10

In [40]:
```
ar = rf_oob_df.plot(legend=False, marker='o', figsize=(10,5))
ar.set(ylabel='out-of-bag error')
```

Out[40]:
```
[Text(0,0.5,'out-of-bag error')]
```



Step11

In [41]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,classification_report
DTC = DecisionTreeClassifier(max_depth=4, random_state=42)
DTC.fit(X_test,y_test)
```

Out[41]:
```
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=4
2,
            splitter='best')
```

In [42]:
```
DTC_y_pred = DTC.predict(X_test)
DTC_y_pred
```

Out[42]:
```
array(['No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'Yes', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', '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', 'N
o',
       'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', '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', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'N
o',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', '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 [43]:
```
from sklearn import tree
from sklearn.tree import export_graphviz
with open("DTC2.dot", 'w') as f:
    f = tree.export_graphviz(DTC,out_file=f,max_depth = 4,impurity = False
        class_names=['Yes', 'No'], filled=True)
```

In [44]:
```
print("Accuracy of test :",DTC.score(X_test,y_test))
```

```
Accuracy of test : 0.9183673469387755
```

In [45]:
```
print(classification_report(y_test,DTC_y_pred))
```

```
             precision    recall  f1-score   support

         No       0.91      1.00      0.96       255
        Yes       1.00      0.38      0.56        39

avg / total       0.93      0.92      0.90       294
```

In [46]:
```
from sklearn.metrics import precision_score, recall_score, accuracy_score,
```

In [47]:
```
from sklearn.metrics import accuracy_score, classification_report
print("Accuracy =", accuracy_score(y_test, y_pred),"\n")
print("Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy = 0.8707482993197279

Report:
              precision    recall  f1-score   support

         No       0.88      0.99      0.93       255
        Yes       0.57      0.10      0.17        39

avg / total       0.84      0.87      0.83       294
```

In [48]:
```
from sklearn.metrics import accuracy_score, classification_report
print("Accuracy =", accuracy_score(y_test, DTC_y_pred),"\n")
print("Report:\n", classification_report(y_test, DTC_y_pred))
```

```
Accuracy = 0.9183673469387755

Report:
              precision    recall  f1-score   support

         No       0.91      1.00      0.96       255
        Yes       1.00      0.38      0.56        39

avg / total       0.93      0.92      0.90       294
```

In [ ]: