225229140 PML LAB 9

In [69]:

```
import pandas as pd
```

In [70]:

```
df=pd.read_csv('Employee_hopping.csv')
df
```

Out[70]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Ed
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	32	No	Travel_Frequently	1005	Research & Development	2	2	
6	59	No	Travel_Rarely	1324	Research & Development	3	3	

In [71]:

```
df.head()
```

Out[71]:

	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 [72]:
```
df.head
```

Out[72]:
```
<bound method NDFrame.head of       Age Attrition     BusinessTravel  DailyRa
te              Department  \
0      41       Yes      Travel_Rarely       1102                   Sales
1      49        No  Travel_Frequently        279  Research & Development
2      37       Yes      Travel_Rarely       1373  Research & Development
3      33        No  Travel_Frequently       1392  Research & Development
4      27        No      Travel_Rarely        591  Research & Development
5      32        No  Travel_Frequently       1005  Research & Development
6      59        No      Travel_Rarely       1324  Research & Development
7      30        No      Travel_Rarely       1358  Research & Development
8      38        No  Travel_Frequently        216  Research & Development
9      36        No      Travel_Rarely       1299  Research & Development
10     35        No      Travel_Rarely        809  Research & Development
11     29        No      Travel_Rarely        153  Research & Development
12     31        No      Travel_Rarely        670  Research & Development
13     34        No      Travel_Rarely       1346  Research & Development
14     28       Yes      Travel_Rarely        103  Research & Development
15     29        No      Travel_Rarely       1389  Research & Development
16     32        No      Travel_Rarely        334  Research & Development
17     22        No         Non-Travel       1123  Research & Development
```

In [73]:
```
df.shape
```

Out[73]:
```
(1470, 35)
```

In [74]:
```
df.columns
```

Out[74]:
```
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 [75]:
```
df.dtypes
```

Out[75]:
```
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 [76]:
```
df['Department'].value_counts()
```

Out[76]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

In [77]:
```
#step 2
```

In [78]:
```
x=df.drop(['Attrition'],axis=1)
y=df.Attrition
```

In [79]:
```
x
```

22	34	Travel_Rarely	419	Research & Development	7	4	Life Scienc
23	21	Travel_Rarely	391	Research & Development	15	2	Life Scienc
24	34	Travel_Rarely	699	Research & Development	6	1	Medi
25	53	Travel_Rarely	1282	Research & Development	5	3	Oth
26	32	Travel_Frequently	1125	Research & Development	16	1	Life Scienc
27	42	Travel_Rarely	691	Sales	8	4	Marketi
28	44	Travel_Rarely	477	Research & Development	7	4	Medi

In [80]:
```
y.head()
```

Out[80]:
```
0     Yes
1      No
2     Yes
3      No
4      No
Name: Attrition, dtype: object
```

In [81]:
```
y=y.apply(lambda x:1 if x=='Yes' else 0)
y.head()
```

Out[81]:
```
0     1
1     0
2     1
3     0
4     0
Name: Attrition, dtype: int64
```

In [82]:
```
#step 3
```

In [83]:

```
df=pd.get_dummies(df,columns=['BusinessTravel','Department','EducationField','Gen
df.head()
```

Out[83]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	Env
0	41	Yes	1102	1	2	1	1	
1	49	No	279	8	1	1	2	
2	37	Yes	1373	2	2	1	4	
3	33	No	1392	3	4	1	5	
4	27	No	591	2	1	1	7	

5 rows × 56 columns

In [84]:

```
#step4
```

In [85]:

```
X=df.drop(['Attrition'],axis=1)
```

In [86]:

```
x.shape
```

Out[86]:

```
(1470, 34)
```

In [87]:

```
y.shape
```

Out[87]:

```
(1470,)
```

In [88]:

```
#step 5
```

In [92]:

```
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 [93]:

```
from sklearn.ensemble import RandomForestClassifier
RFC = RandomForestClassifier(n_estimators=100, max_features=0.3)
```

In [94]:

```
RFC.fit(x_train,y_train)
```

Out[94]:

```
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 [95]:

```
RFC_y_pred = RFC.predict(x_test)
RFC_y_pred
```

Out[95]:

```
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [96]:

```
#step 6
```

In [97]:

```
from sklearn.metrics import accuracy_score,classification_report
```

In [98]:

```
RFC_acc = accuracy_score(y_test,RFC_y_pred)
RFC_acc
```

Out[98]:

```
0.8741496598639455
```

In [101]:

```
print(classification_report(y_test,RFC_y_pred))
```

```
             precision    recall  f1-score   support

          0       0.88      0.99      0.93       255
          1       0.62      0.13      0.21        39

avg / total       0.85      0.87      0.84       294
```

In [102]:

```
#step 7
```

In [103]:

```
print(RFC.feature_importances_)
```

```
[0.05681829 0.04803636 0.04079115 0.01536416 0.         0.04211821
 0.0223703  0.03940114 0.01953665 0.02639835 0.02068489 0.07963548
 0.04391964 0.03136154 0.02727713 0.00297495 0.01685568 0.
 0.02960626 0.05055773 0.02387794 0.01912027 0.0362398  0.02830433
 0.02176011 0.0268212  0.00315349 0.01360907 0.00552435 0.00179344
 0.00685234 0.00863051 0.00215007 0.00442699 0.00563969 0.00458001
 0.00279996 0.00708596 0.00606265 0.00549243 0.00138097 0.00325285
 0.00813727 0.00084306 0.00221952 0.00066364 0.00519339 0.00745658
 0.00666573 0.00420261 0.0057582  0.02059756 0.         0.05182117
 0.0341749 ]
```

In [106]:

```
feature_name = pd.DataFrame(RFC.feature_importances_, index=x_train.columns, colu
feature_name
```

```
               EducationField_Medical   0.004580
                 EducationField_Other   0.002800
      EducationField_Technical Degree   0.007086
                        Gender_Female   0.006063
                          Gender_Male   0.005492
  JobRole_Healthcare Representative     0.001381
           JobRole_Human Resources      0.003253
      JobRole_Laboratory Technician     0.008137
                    JobRole_Manager     0.000843
     JobRole_Manufacturing Director     0.002220
         JobRole_Research Director      0.000664
        JobRole_Research Scientist      0.005193
           JobRole_Sales Executive      0.007457
```

In [ ]:

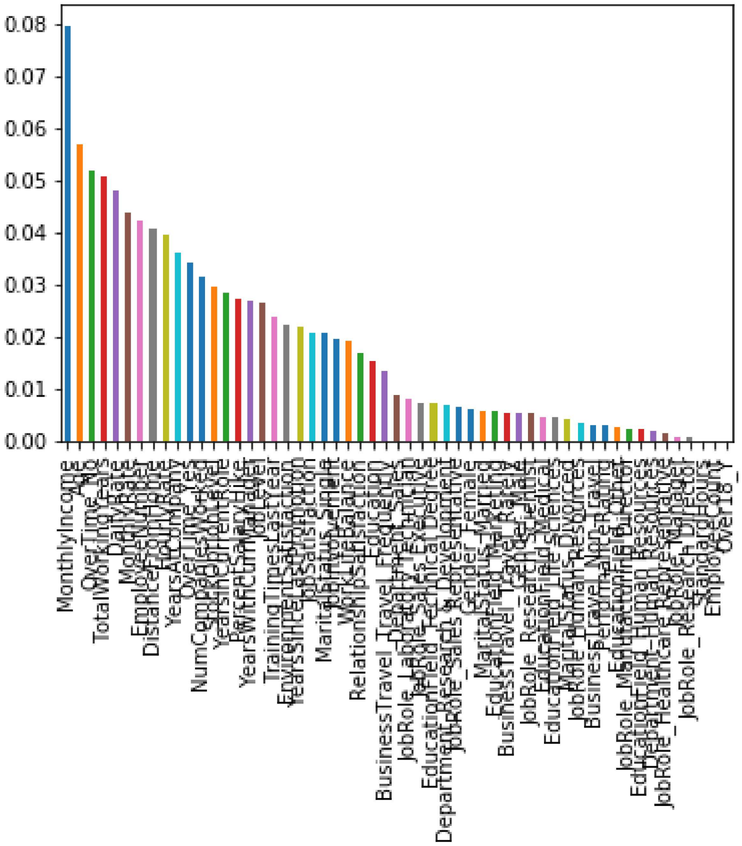
```

```

In [110]:

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

```
In [114]: pd.Series(RFC.feature_importances_, index=x_train.columns).sort_values(ascending=
```

```
Out[114]: <matplotlib.axes._subplots.AxesSubplot at 0x1f129362cf8>
```



```
In [115]: #step 8
```

In [121]:

```
estimator=RFC.estimators_[5]
```

In [125]:

```
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,fe
```

In [126]:

```
!dot-Tpng RFDT.dot -o RFDT.png
```

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

In [ ]:

In [130]:

```
#step 9
```

In [138]:

```
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
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\ensemble\forest.py:453: UserWarning: Some inputs do not have OOB
scores. This probably means too few trees were used to compute any reliable oob
estimates.
  warn("Some inputs do not have OOB scores. "
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\ensemble\forest.py:458: RuntimeWarning: invalid value encountered
in true_divide
  predictions[k].sum(axis=1)[:, np.newaxis])
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\ensemble\forest.py:453: UserWarning: Some inputs do not have OOB
scores. This probably means too few trees were used to compute any reliable oob
estimates.
  warn("Some inputs do not have OOB scores. "
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Anaconda3_64\lib\site-pac
kages\sklearn\ensemble\forest.py:458: RuntimeWarning: invalid value encountered
in true_divide
  predictions[k].sum(axis=1)[:, np.newaxis])
```

Out[138]:

	oob
n_trees	
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

In [139]:

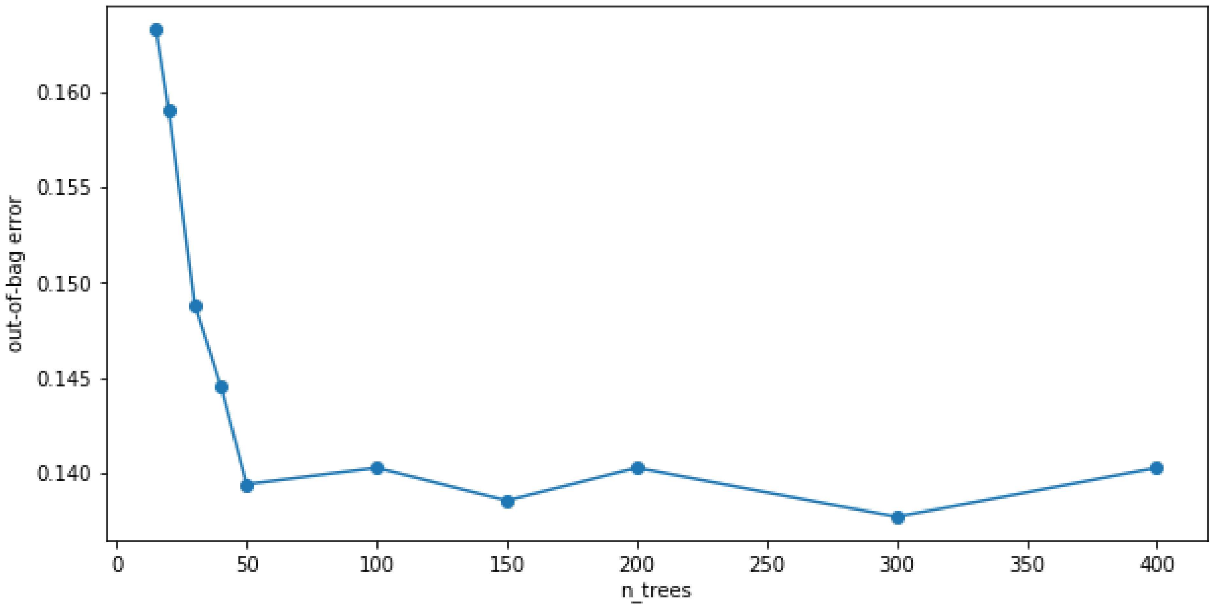
```
#step 10
```

In [140]:

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

Out[140]:

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



In [141]:

```
#step 11
```

In [143]:

```
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[143]:

```
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 [145]: y_pred1 = clf.predict(x_test)
          y_pred1
```

```
Out[145]: array([0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
                 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

```
In [147]: ort tree
          e import export_graphviz
          dot", 'w') as f:
          ort_graphviz(clf,out_file=f,max_depth = 4,impurity = False,feature_names =X.column
```

```
In [148]: !dot -Tpng DTC2.dot -o DTC2.png
```

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

```
In [150]: print("Accuracy of test :",clf.score(x_test,y_test))
```

```
Accuracy of test : 0.9183673469387755
```

```
In [151]: print(classification_report(y_test,RFC_y_pred))
```

```
             precision    recall  f1-score   support

          0       0.88      0.99      0.93       255
          1       0.62      0.13      0.21        39

avg / total       0.85      0.87      0.84       294
```

```
In [153]: from sklearn.metrics import precision_score, recall_score, accuracy_score, roc_au
```

In [155]:

```
print("RF model :",accuracy_score(y_test,RFC_y_pred))
print("RF Precision:",precision_score(y_test,RFC_y_pred))
print("RF Recall :",recall_score(y_test,RFC_y_pred))
print("RF F1 score :",f1_score(y_test,RFC_y_pred))
print("\n")
print("DT model :",accuracy_score(y_test,y_pred1))
print("DT Precision:",precision_score(y_test,y_pred1))
print("DT Recall :",recall_score(y_test,y_pred1))
print("DT F1 score :",f1_score(y_test,y_pred1))
```

```
RF model : 0.8741496598639455
RF Precision: 0.625
RF Recall : 0.1282051282051282
RF F1 score : 0.21276595744680848


DT model : 0.9183673469387755
DT Precision: 1.0
DT Recall : 0.38461538461538464
DT F1 score : 0.5555555555555556
```

In [ ]: