

▼ Assignment-04 Sep 22

M VIVEK(21BCE9526)

1.Download the Employee Attrition Dataset <https://www.kaggle.com/datasets/patelprashant/employee-attrition>

2.Perfrom Data Preprocessing

3.Model Building using Logistic Regression and Decision Tree

4.Calculate Performance metrics

```
import numpy as np
import pandas as pd
import matplotlib as plt
import seaborn as sns
```

```
df = pd.read_csv('/content/drive/MyDrive/SmartInternz-Notebooks/HR-Employee-Attrition.csv')
```

```
df
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7
...
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical	1	2061
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical	1	2062
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	1	2064
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical	1	2065
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical	1	2068

1470 rows × 35 columns

```
df.shape
```

```
(1470, 35)
```

```
df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	.
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	.
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	.
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	.
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	.

5 rows × 35 columns

```
df.BusinessTravel.value_counts()
```

Travel_Rarely	1043
Travel_Frequently	277
Non-Travel	150
Name: BusinessTravel, dtype: int64	

```
df.Department.value_counts()
```

Research & Development	961
Sales	446
Human Resources	63
Name: Department, dtype: int64	

```
df.info()
```

<class 'pandas.core.frame.DataFrame'>	
RangeIndex: 1470 entries, 0 to 1469	
Data columns (total 35 columns):	
# Column	Non-Null Count Dtype
---	-----
0 Age	1470 non-null int64
1 Attrition	1470 non-null object
2 BusinessTravel	1470 non-null object
3 DailyRate	1470 non-null int64
4 Department	1470 non-null object
5 DistanceFromHome	1470 non-null int64
6 Education	1470 non-null int64
7 EducationField	1470 non-null object
8 EmployeeCount	1470 non-null int64
9 EmployeeNumber	1470 non-null int64
10 EnvironmentSatisfaction	1470 non-null int64
11 Gender	1470 non-null object
12 HourlyRate	1470 non-null int64
13 JobInvolvement	1470 non-null int64
14 JobLevel	1470 non-null int64
15 JobRole	1470 non-null object
16 JobSatisfaction	1470 non-null int64
17 MaritalStatus	1470 non-null object
18 MonthlyIncome	1470 non-null int64
19 MonthlyRate	1470 non-null int64
20 NumCompaniesWorked	1470 non-null int64
21 Over18	1470 non-null object
22 Overtime	1470 non-null object
23 PercentSalaryHike	1470 non-null int64
24 PerformanceRating	1470 non-null int64
25 RelationshipSatisfaction	1470 non-null int64
26 StandardHours	1470 non-null int64
27 StockOptionLevel	1470 non-null int64
28 TotalWorkingYears	1470 non-null int64
29 TrainingTimesLastYear	1470 non-null int64
30 WorkLifeBalance	1470 non-null int64
31 YearsAtCompany	1470 non-null int64
32 YearsInCurrentRole	1470 non-null int64
33 YearsSinceLastPromotion	1470 non-null int64
34 YearsWithCurrManager	1470 non-null int64

```
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

```
df.describe()
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobIn
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000		1470.000000	1470.000000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306		2.721769	65.891156
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335		1.093082	20.329428
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000		1.000000	30.000000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000		2.000000	48.000000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000		3.000000	66.000000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000		4.000000	83.750000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000		4.000000	100.000000

8 rows × 26 columns

```
#Dropping the unwanted columns
```

```
df.drop(['JobSatisfaction','JobInvolvement','Over18','RelationshipSatisfaction','EnvironmentSatisfaction','DistanceFromHome','StockOptionLeve
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 28 columns):
 #   Column           Non-Null Count  Dtype  
 ---  --  
 0   Age              1470 non-null   int64  
 1   Attrition        1470 non-null   object 
 2   BusinessTravel   1470 non-null   object 
 3   DailyRate         1470 non-null   int64  
 4   Department        1470 non-null   object 
 5   Education         1470 non-null   int64  
 6   EducationField    1470 non-null   object 
 7   EmployeeCount     1470 non-null   int64  
 8   EmployeeNumber    1470 non-null   int64  
 9   Gender            1470 non-null   object 
 10  HourlyRate        1470 non-null   int64  
 11  JobLevel          1470 non-null   int64  
 12  JobRole           1470 non-null   object 
 13  MaritalStatus     1470 non-null   object 
 14  MonthlyIncome     1470 non-null   int64  
 15  MonthlyRate       1470 non-null   int64  
 16  NumCompaniesWorked 1470 non-null   int64  
 17  Overtime          1470 non-null   object 
 18  PercentSalaryHike 1470 non-null   int64  
 19  PerformanceRating 1470 non-null   int64  
 20  StandardHours     1470 non-null   int64  
 21  TotalWorkingYears 1470 non-null   int64  
 22  TrainingTimesLastYear 1470 non-null   int64  
 23  WorkLifeBalance   1470 non-null   int64  
 24  YearsAtCompany    1470 non-null   int64  
 25  YearsInCurrentRole 1470 non-null   int64  
 26  YearsSinceLastPromotion 1470 non-null   int64  
 27  YearsWithCurrManager 1470 non-null   int64  
dtypes: int64(20), object(8)
memory usage: 321.7+ KB
```

```
df.isnull().any()
```

Age	False
Attrition	False
BusinessTravel	False
DailyRate	False
Department	False
Education	False
EducationField	False
EmployeeCount	False
EmployeeNumber	False
Gender	False
HourlyRate	False
JobLevel	False

```

JobRole           False
MaritalStatus     False
MonthlyIncome     False
MonthlyRate       False
NumCompaniesWorked False
OverTime          False
PercentSalaryHike False
PerformanceRating False
StandardHours     False
TotalWorkingYears False
TrainingTimesLastYear False
WorkLifeBalance   False
YearsAtCompany    False
YearsInCurrentRole False
YearsSinceLastPromotion False
YearsWithCurrManager False
dtype: bool

```

```
df.isnull().sum()
```

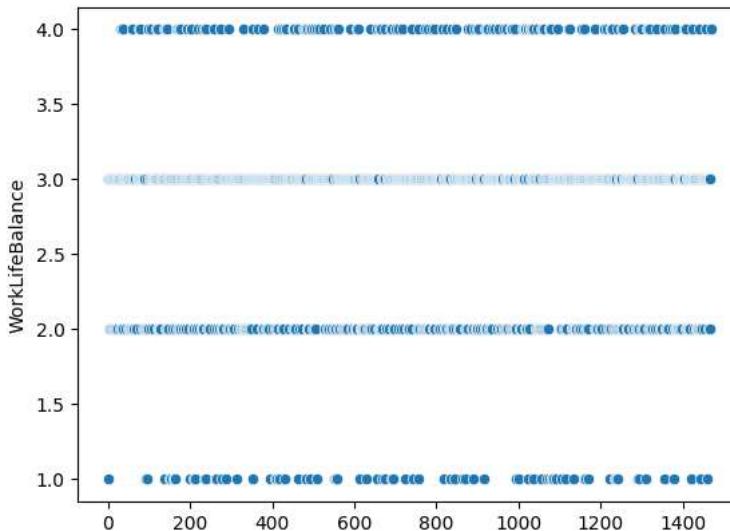
```

Age                  0
Attrition            0
BusinessTravel        0
DailyRate             0
Department           0
Education             0
EducationField         0
EmployeeCount          0
EmployeeNumber         0
Gender                0
HourlyRate            0
JobLevel              0
JobRole               0
MaritalStatus          0
MonthlyIncome          0
MonthlyRate            0
NumCompaniesWorked     0
OverTime               0
PercentSalaryHike       0
PerformanceRating       0
StandardHours          0
TotalWorkingYears        0
TrainingTimesLastYear    0
WorkLifeBalance          0
YearsAtCompany          0
YearsInCurrentRole       0
YearsSinceLastPromotion    0
YearsWithCurrManager      0
dtype: int64

```

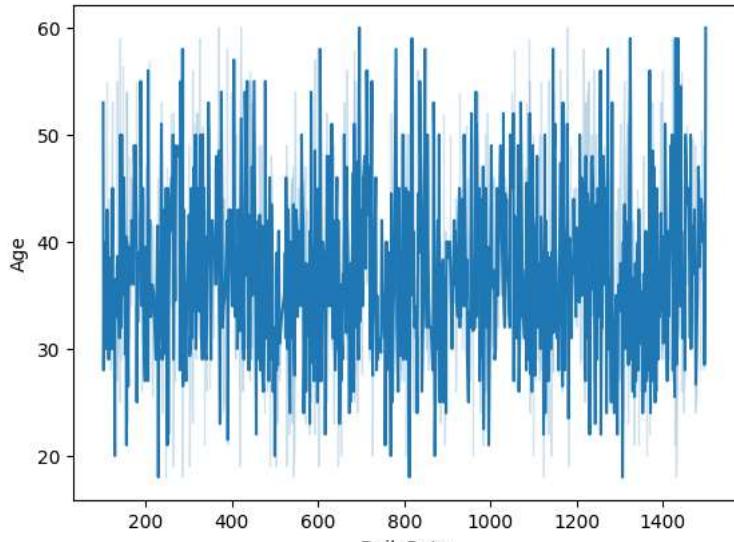
```
sns.scatterplot(df['WorkLifeBalance'])
```

```
<Axes: ylabel='WorkLifeBalance'>
```



```
sns.lineplot(x='DailyRate',y='Age',data=df)
```

```
<Axes: xlabel='DailyRate', ylabel='Age'>
```



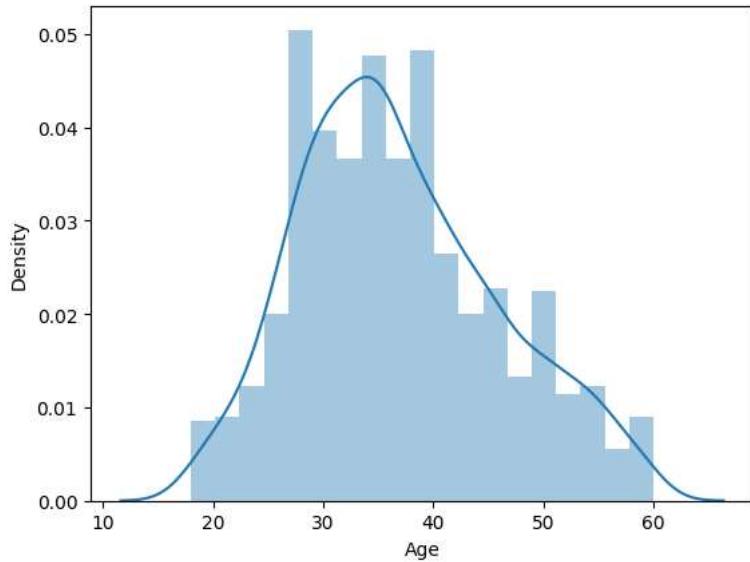
```
sns.distplot(df['Age'])
```

```
<ipython-input-16-0faf04ea3f6>:1: UserWarning:  
  'distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
<https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(df['Age'])  
<Axes: xlabel='Age', ylabel='Density'>
```



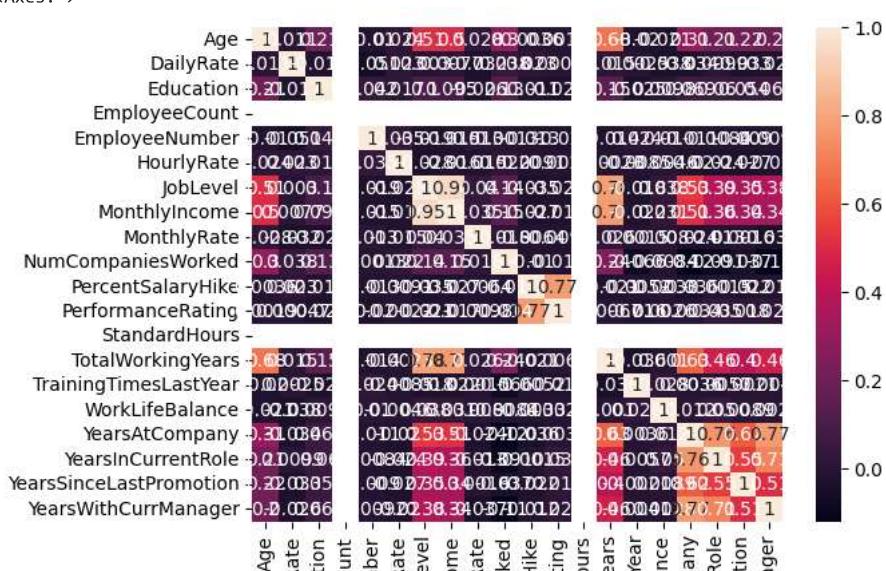
```
df.corr()
```

<ipython-input-17-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version df.corr()

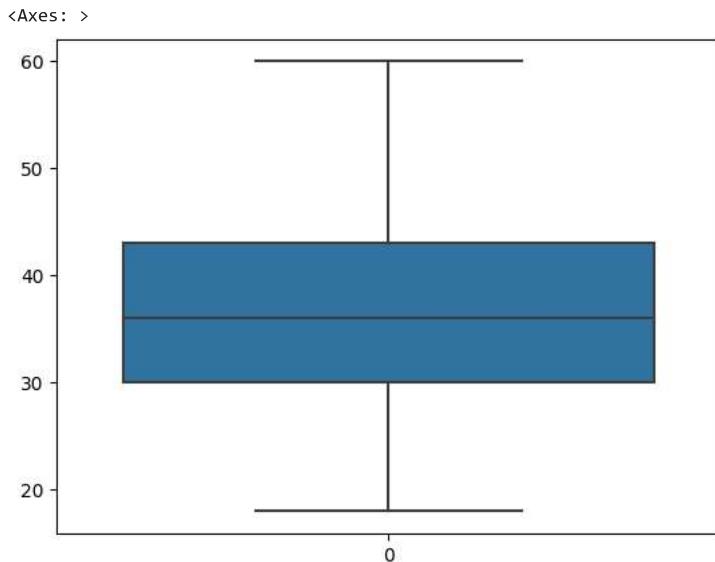
	Age	DailyRate	Education	EmployeeCount	EmployeeNumber	HourlyRate	JobLevel	MonthlyIncome	MonthlyRate
Age	1.000000	0.010661	0.208034	NaN	-0.010145	0.024287	0.509604	0.497855	0.028051
DailyRate	0.010661	1.000000	-0.016806	NaN	-0.050990	0.023381	0.002966	0.007707	-0.032182
Education	0.208034	-0.016806	1.000000	NaN	0.042070	0.016775	0.101589	0.094961	-0.026084
EmployeeCount	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
EmployeeNumber	-0.010145	-0.050990	0.042070	NaN	1.000000	0.035179	-0.018519	-0.014829	0.012648
HourlyRate	0.024287	0.023381	0.016775	NaN	0.035179	1.000000	-0.027853	-0.015794	-0.015297
JobLevel	0.509604	0.002966	0.101589	NaN	-0.018519	-0.027853	1.000000	0.950300	0.039563
MonthlyIncome	0.497855	0.007707	0.094961	NaN	-0.014829	-0.015794	0.950300	1.000000	0.034814
MonthlyRate	0.028051	-0.032182	-0.026084	NaN	0.012648	-0.015297	0.039563	0.034814	1.000000
NumCompaniesWorked	0.299635	0.038153	0.126317	NaN	-0.001251	0.022157	0.142501	0.149515	0.017521
PercentSalaryHike	0.003634	0.022704	-0.011111	NaN	-0.012944	-0.009062	-0.034730	-0.027269	-0.006429
PerformanceRating	0.001904	0.000473	-0.024539	NaN	-0.020359	-0.002172	-0.021222	-0.017120	-0.009811
StandardHours	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
TotalWorkingYears	0.680381	0.014515	0.148280	NaN	-0.014365	-0.002334	0.782208	0.772893	0.026442
TrainingTimesLastYear	-0.019621	0.002453	-0.025100	NaN	0.023603	-0.008548	-0.018191	-0.021736	0.001467
WorkLifeBalance	-0.021490	-0.037848	0.009819	NaN	0.010309	-0.004607	0.037818	0.030683	0.007963
YearsAtCompany	0.311309	-0.034055	0.069114	NaN	-0.011240	-0.019582	0.534739	0.514285	-0.023655
YearsInCurrentRole	0.212901	0.009932	0.060236	NaN	-0.008416	-0.024106	0.389447	0.363818	-0.012815
YearsSinceLastPromotion	0.216513	-0.033229	0.054254	NaN	-0.009019	-0.026716	0.353885	0.344978	0.001567
YearsWithCurrManager	0.202089	-0.026363	0.069065	NaN	-0.009197	-0.020123	0.375281	0.344079	-0.036746

```
sns.heatmap(df.corr(), annot=True)
```

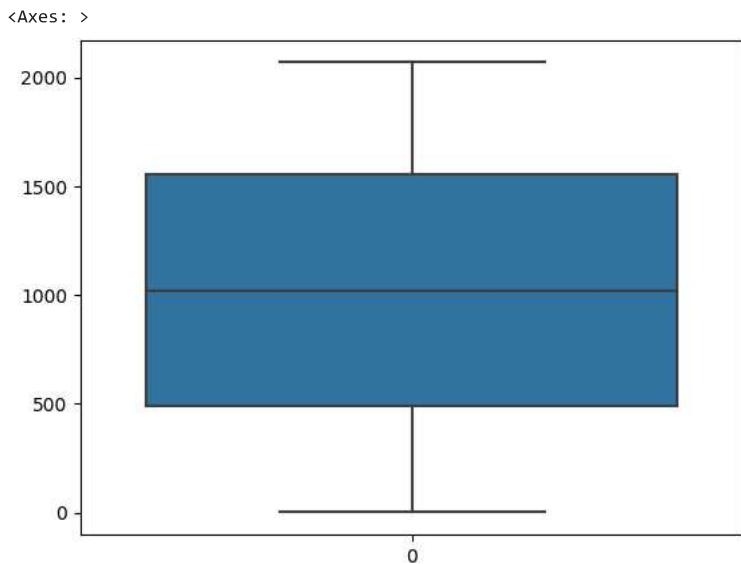
```
<ipython-input-18-8df7bcac526d>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version
  sns.heatmap(df.corr(), annot=True)
<Axes: >
```



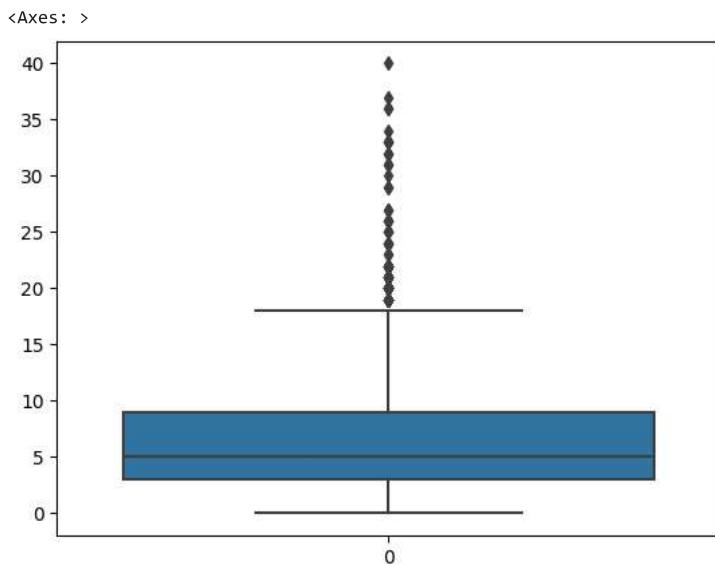
```
sns.boxplot(df['Age'])
```



```
sns.boxplot(df['EmployeeNumber'])
```



```
sns.boxplot(df['YearsAtCompany'])
```



```
q1=df.YearsAtCompany.quantile(0.25)
q3=df.YearsAtCompany.quantile(0.75)
```

```
q1,q3
```

```
(3.0, 9.0)
```

```
IQR=q3-q1
```

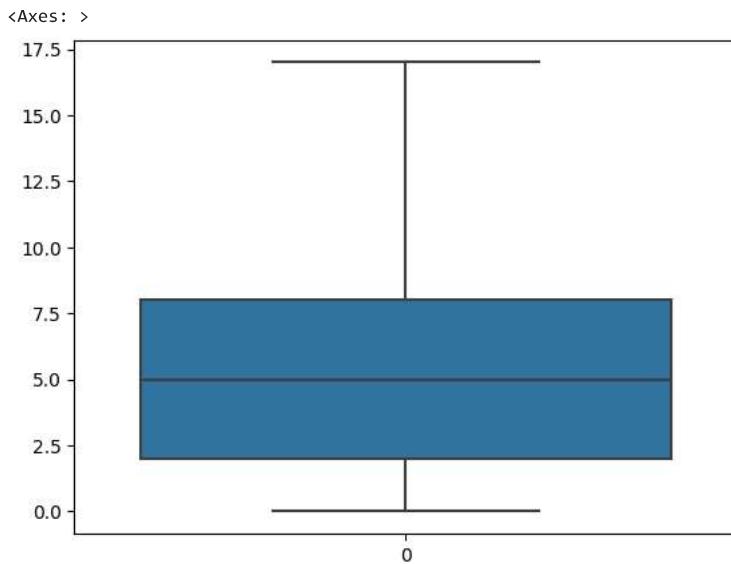
```
upper_limit = q3+1.5*IQR
```

```
upper_limit
```

```
18.0
```

```
df=df[df['YearsAtCompany']<upper_limit]
```

```
sns.boxplot(df['YearsAtCompany'])
```



```
df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	Education	EducationField	EmployeeCount	EmployeeNumber	Gender	...	Percent
0	41	Yes	Travel_Rarely	1102	Sales	2	Life Sciences	1	1	Female	...	
1	49	No	Travel_Frequently	279	Research & Development	1	Life Sciences	1	2	Male	...	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	Other	1	4	Male	...	
3	33	No	Travel_Frequently	1392	Research & Development	4	Life Sciences	1	5	Female	...	
4	27	No	Travel_Rarely	591	Research & Development	1	Medical	1	7	Male	...	

5 rows × 28 columns

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

x.head()

	Age	BusinessTravel	DailyRate	Department	Education	EducationField	EmployeeCount	EmployeeNumber	Gender	HourlyRate	...	Percent
0	41	Travel_Rarely	1102	Sales	2	Life Sciences	1	1	Female	94	...	
1	49	Travel_Frequently	279	Research & Development	1	Life Sciences	1	2	Male	61	...	
2	37	Travel_Rarely	1373	Research & Development	2	Other	1	4	Male	92	...	
3	33	Travel_Frequently	1392	Research & Development	4	Life Sciences	1	5	Female	56	...	
4	27	Travel_Rarely	591	Research & Development	1	Medical	1	7	Male	40	...	

5 rows × 27 columns

y=df.Attrition

y.head()

```

0    Yes
1     No
2    Yes
3     No
4     No
Name: Attrition, dtype: object

```

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

```

columns=['BusinessTravel','Department','EducationField','Gender','JobRole','MaritalStatus','OverTime']
x[columns]=x[columns].apply(le.fit_transform)

```

x.head()

	Age	BusinessTravel	DailyRate	Department	Education	EducationField	EmployeeCount	EmployeeNumber	Gender	HourlyRate	...	Percent
0	41	2	1102	2	2	1	1	1	0	94	...	
1	49	1	279	1	1	1	1	2	1	61	...	
2	37	2	1373	1	2	4	1	4	1	92	...	
3	33	1	1392	1	4	1	1	5	0	56	...	
4	27	2	591	1	1	3	1	7	1	40	...	

5 rows × 27 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1353 entries, 0 to 1469
Data columns (total 28 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Age              1353 non-null    int64  
 1   Attrition        1353 non-null    object  
 2   BusinessTravel   1353 non-null    object  
 3   DailyRate        1353 non-null    int64  
 4   Department       1353 non-null    object  
 5   Education        1353 non-null    int64  
 6   EducationField   1353 non-null    object  
 7   EmployeeCount    1353 non-null    int64  
 8   EmployeeNumber   1353 non-null    int64  
 9   Gender            1353 non-null    object  
 10  HourlyRate       1353 non-null    int64  
 11  JobLevel         1353 non-null    int64  
 12  JobRole          1353 non-null    object  
 13  MaritalStatus    1353 non-null    object  
 14  MonthlyIncome    1353 non-null    int64  
 15  MonthlyRate      1353 non-null    int64  
 16  NumCompaniesWorked 1353 non-null    int64  
 17  Overtime          1353 non-null    object  
 18  PercentSalaryHike 1353 non-null    int64  
 19  PerformanceRating 1353 non-null    int64  
 20  StandardHours    1353 non-null    int64  
 21  TotalWorkingYears 1353 non-null    int64  
 22  TrainingTimesLastYear 1353 non-null    int64  
 23  WorkLifeBalance   1353 non-null    int64  
 24  YearsAtCompany    1353 non-null    int64  
 25  YearsInCurrentRole 1353 non-null    int64  
 26  YearsSinceLastPromotion 1353 non-null    int64  
 27  YearsWithCurrManager 1353 non-null    int64  
dtypes: int64(20), object(8)
memory usage: 338.8+ KB
```

```
print(le.classes_)
```

```
['No' 'Yes']
```

```
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
```

```
x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
```

```
x_scaled.head()
```

	Age	BusinessTravel	DailyRate	Department	Education	EducationField	EmployeeCount	EmployeeNumber	Gender	HourlyRate	...	Per
0	0.547619	1.0	0.716332	1.0	0.25	0.2	0.0	0.000000	0.0	0.914286	...	
1	0.738095	0.5	0.126791	0.5	0.00	0.2	0.0	0.000484	1.0	0.442857	...	
2	0.452381	1.0	0.910458	0.5	0.25	0.8	0.0	0.001451	1.0	0.885714	...	
3	0.357143	0.5	0.924069	0.5	0.75	0.2	0.0	0.001935	0.0	0.371429	...	
4	0.214286	1.0	0.350287	0.5	0.00	0.6	0.0	0.002903	1.0	0.142857	...	

5 rows × 27 columns

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test = train_test_split(x_scaled,y,test_size=0.2,random_state=0)
```

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((1082, 27), (271, 27), (1082,), (271,))
```

Model building

▼ Logistic Regression

```
from sklearn.linear_model import LogisticRegression  
model=LogisticRegression()
```

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

▼ LogisticRegression

```
pred=model.predict(x_test)
```

pred

y_test

```
459      No
1076     No
1377     No
91       No
1103     No
...
525      Yes
1159     No
1179     No
1435     No
531      No
Name: Attrition, Length: 271, dtype: object
```

```
model.predict(ms.transform([[41,2,1102,2,2,1,1,1,0,94,2,4,3,1,3,1,1,11,3,80,8,0,1,6,4,0,5]]))
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but MinMaxScaler was fitted
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was fitted
  warnings.warn(
array(['Yes'], dtype=object)
```

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

```
print(accuracy score(y test,pred))
```

0.8560885608856088

```
print(classification_report(y_test,pred))
```

	precision	recall	f1-score	support
No	0.87	0.97	0.92	228
Yes	0.62	0.23	0.34	43
accuracy			0.86	271
macro avg	0.75	0.60	0.63	271
weighted avg	0.83	0.86	0.83	271

```
pd.crosstab(y_test,pred)
```

	col_0	No	Yes	
Attrition				
No	222	6		
Yes	33	10		

▼ Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier  
dtc=DecisionTreeClassifier()
```

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

```
    ▾ DecisionTreeClassifier  
DecisionTreeClassifier()
```

```
y_pred=dtc.predict(x_test)
```

y_pred

y_test

459	No
1076	No
1377	No
91	No
1103	No

```

525    Yes
1159   No
1179   No
1435   No
531    No
Name: Attrition, Length: 271, dtype: object

```

```
dtc.predict(ms.transform([[41,2,1102,2,2,1,1,1,0,94,2,4,3,1,3,1,1,11,3,80,8,0,1,6,4,0,5]]))
```

```

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but MinMaxScaler was fitted
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted
  warnings.warn(
array(['Yes'], dtype=object)

```

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

```
print(accuracy_score(y_test, y_pred))
```

```
0.7601476014760148
```

```
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
No	0.87	0.84	0.85	228
Yes	0.29	0.35	0.32	43
accuracy			0.76	271
macro avg	0.58	0.59	0.59	271
weighted avg	0.78	0.76	0.77	271

▼ Hyper parameter tuning

```

from sklearn.model_selection import GridSearchCV
parameter={
  'criterion':['gini','entropy'],
  'splitter':['best','random'],
  'max_depth':[1,2,3,4,5],
  'max_features':['auto', 'sqrt', 'log2']
}

grid_search=GridSearchCV(estimator=dtc, param_grid=parameter, cv=5, scoring="accuracy")

grid_search.fit(x_train, y_train)

```


`grid_search.best_params_`

```
{'criterion': 'gini',
 'max_depth': 4,
 'max_features': 'sqrt',
 'splitter': 'random'}
```

```
dtc_cv=DecisionTreeClassifier(criterion= 'gini',
    max_depth=4,
    max_features='sqrt',
    splitter='random')
dtc_cv.fit(x_train,y_train)
```

```
DecisionTreeClassifier(max_depth=4, max_features=
```

```
y1_pred=dtc_cv.predict(x_test)
```

```
print(accuracy_score(y_test,y1_pred))
```

0.8302583025830258

```
print(classification_report(y_test,y1_pred))
```

precision	recall	f1-score	support
0.85	0.97	0.91	228
0.36	0.09	0.15	43

accuracy		0.83	271
macro avg	0.61	0.53	271
weighted avg	0.77	0.83	0.79

▼ Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()

rfc.fit(x_train,y_train)

y2_pred=rfc.predict(x_test)

y2_pred

array(['No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', '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', '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', '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'],
      dtype=object)
```

```
y_test

459    No
1076   No
1377   No
91     No
1103   No
...
525    Yes
1159   No
1179   No
1435   No
531    No
Name: Attrition, Length: 271, dtype: object
```

```
print(accuracy_score(y_test,y2_pred))
```

```
0.8560885608856088
```

```
print(classification_report(y_test,y2_pred))
```

	precision	recall	f1-score	support
No	0.87	0.98	0.92	228
Yes	0.64	0.21	0.32	43
accuracy			0.86	271

macro avg	0.76	0.59	0.62	271
weighted avg	0.83	0.86	0.82	271

```
forest_params = [{'max_depth': list(range(10, 15)), 'max_features': list(range(0,14))}]
```

```
rfc_cv= GridSearchCV(rfc,param_grid=forest_params, cv=10,scoring="accuracy")
```

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

```
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_validation.py:378: FitFailedWarning:  
50 fits failed out of a total of 700.  
The score on these train-test partitions for these parameters will be set to nan.  
If these failures are not expected, you can try to debug them by setting error_score='raise'.
```

Below are more details about the failures:

50 fits failed with the following error:

Traceback (most recent call last):

```
File "/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_validation.py", line 686, in _fit_and_score  
    estimator.fit(X_train, y_train, **fit_params)  
File "/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_forest.py", line 340, in fit  
    self._validate_params()  
File "/usr/local/lib/python3.10/dist-packages/sklearn/base.py", line 600, in _validate_params  
    validate_parameter_constraints()  
File "/usr/local/lib/python3.10/dist-packages/sklearn/utils/_param_validation.py", line 97, in validate_parameter_constraints  
    raise InvalidParameterError(  
sklearn.utils._param_validation.InvalidParameterError: The 'max_features' parameter of RandomForestClassifier must be an int in the rang  
  
    warnings.warn(some_fits_failed_message, FitFailedWarning)  
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_search.py:952: UserWarning: One or more of the test scores are non-fini  
0.84470778 0.84378186 0.84563371 0.84380734 0.84471628 0.85488447  
0.84562521 0.85027183      nan 0.83734285 0.84288141 0.84194699  
0.84285593 0.84472477 0.84470778 0.8456507 0.8483945 0.84195549  
0.84657662 0.84841148 0.84193      0.8456507      nan 0.83825178  
0.83733435 0.8456422 0.83824329 0.8419385 0.84472477 0.8391947  
0.84563371 0.84380734 0.84471628 0.84746857 0.85210669 0.84751104  
      nan 0.83825178 0.84286442 0.8456422 0.84656813 0.84933741  
0.84655963 0.84751104 0.84565919 0.84657662 0.84563371 0.85210669  
0.84655963 0.85211519      nan 0.83826028 0.83826028 0.8456422  
0.84102107 0.84840299 0.84750255 0.8456507 0.84469079 0.85025484  
0.84749405 0.84655114 0.8520982 0.84751954]  
    warnings.warn(  
    GridSearchCV  
    estimator: RandomForestClassifier  
        RandomForestClassifier
```

```
y3_pred=rfc_cv.predict(x_test)
```

```
print(accuracy_score(y_test,y3_pred))
```

```
0.8376383763837638
```

```
print(classification_report(y_test,y3_pred))
```

	precision	recall	f1-score	support
No	0.87	0.95	0.91	228
Yes	0.48	0.23	0.31	43
accuracy			0.84	271
macro avg	0.67	0.59	0.61	271
weighted avg	0.81	0.84	0.81	271

```
rfc_cv.best_params_
```

```
{'max_depth': 10, 'max_features': 11}
```

