ASSIGNMENT-4 Sept 22

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

2.Perform Data Preprocessing

3.Model Building using Logistic Regression and Decision Tree and Random Forest

4.Calculate Performance metrics

• Data Preprocessing.

```
o    Import the Libraries.
o    Importing the dataset.
o    Checking for Null Values.
o    Data Visualization.
o    Outlier Detection
o    Splitting Dependent and Independent variables
o-    Encoding
o    Feature Scaling.
o    Splitting Data into Train and Test.
```

```
#Import the Libraries.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
#Importing the dataset.
df=pd.read_csv("/content/WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

```
df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	YearsInCurrent
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	80	0	8	0	1	6	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	80	1	10	3	3	10	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2	80	0	7	3	3	0	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3	80	0	8	3	3	8	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4	80	1	6	3	3	2	

5 rows × 35 columns

```
df.shape
```

```
(1470, 35)
```

```
df.StockOptionLevel.value_counts()
```

```
0    631
1    596
2    158
3     85
Name: StockOptionLevel, dtype: int64
```

```
df.EmployeeCount.value_counts()
```

```
1    1470
Name: EmployeeCount, 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	JobInvolvement	JobLevel	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	Years
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	...	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	14
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.891156	2.729932	2.063946	...	2.712245	80.0	0.793878	11.279592	2.799320	2.761224	
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.329428	0.711561	1.106940	...	1.081209	0.0	0.852077	7.780782	1.289271	0.706476	
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	1.000000	1.000000	...	1.000000	80.0	0.000000	0.000000	0.000000	1.000000	
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.000000	2.000000	1.000000	...	2.000000	80.0	0.000000	6.000000	2.000000	2.000000	
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.000000	3.000000	2.000000	...	3.000000	80.0	1.000000	10.000000	3.000000	3.000000	
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.750000	3.000000	3.000000	...	4.000000	80.0	1.000000	15.000000	3.000000	3.000000	
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.000000	4.000000	5.000000	...	4.000000	80.0	3.000000	40.000000	6.000000	4.000000	

8 rows × 26 columns

```
#Checking for Null Values.
df.isnull().any()
```

```
Age                         False
Attrition                   False
BusinessTravel              False
DailyRate                   False
Department                  False
DistanceFromHome            False
Education                   False
EducationField              False
EmployeeCount               False
EmployeeNumber              False
EnvironmentSatisfaction     False
Gender                      False
HourlyRate                  False
JobInvolvement              False
JobLevel                    False
JobRole                     False
JobSatisfaction             False
MaritalStatus               False
MonthlyIncome               False
MonthlyRate                 False
NumCompaniesWorked          False
Over18                      False
OverTime                    False
PercentSalaryHike           False
PerformanceRating           False
RelationshipSatisfaction    False
StandardHours               False
StockOptionLevel            False
```

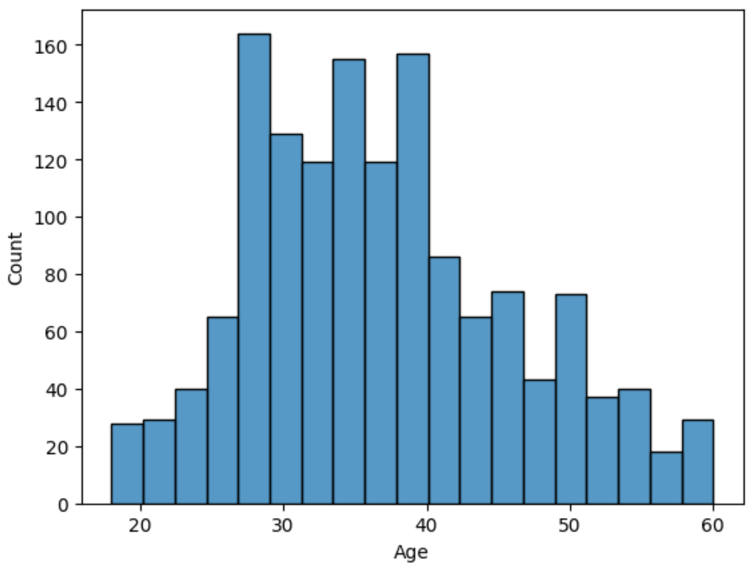
```
TotalWorkingYears           False
TrainingTimesLastYear       False
WorkLifeBalance             False
YearsAtCompany              False
YearsInCurrentRole          False
YearsSinceLastPromotion     False
YearsWithCurrManager        False
dtype: bool
```

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

```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

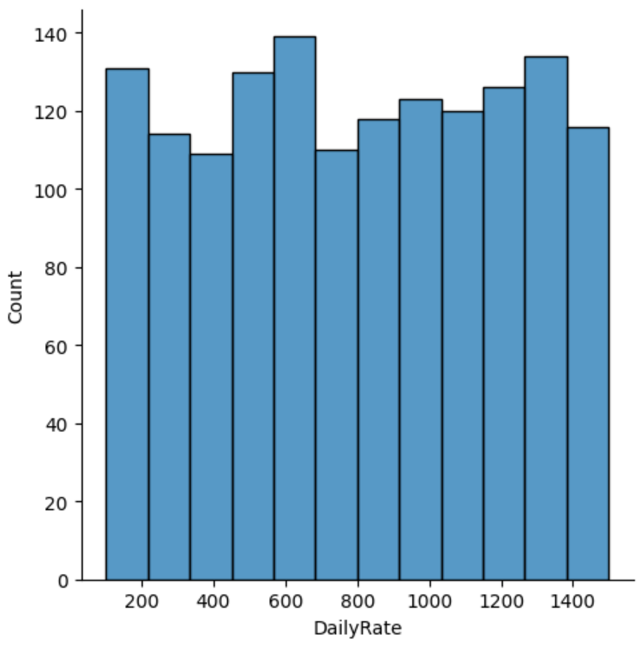
```
#Data Visualization.
sns.histplot(df["Age"])
```

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




```
#Data Visualization.
sns.displot(df["DailyRate"])
```

```
<seaborn.axisgrid.FacetGrid at 0x7cfeafca8190>
```




```
df.corr()
```

```
<ipython-input-16-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalan
Age	1.000000	0.010661	-0.001686	0.208034	NaN	-0.010145	0.010146	0.024287	0.029820	0.509604	...	0.053535	NaN	0.037510	0.680381	-0.019621	-0.0214
DailyRate	0.010661	1.000000	-0.004985	-0.016806	NaN	-0.050990	0.018355	0.023381	0.046135	0.002966	...	0.007846	NaN	0.042143	0.014515	0.002453	-0.0378
DistanceFromHome	-0.001686	-0.004985	1.000000	0.021042	NaN	0.032916	-0.016075	0.031131	0.008783	0.005303	...	0.006557	NaN	0.044872	0.004628	-0.036942	-0.0265
Education	0.208034	-0.016806	0.021042	1.000000	NaN	0.042070	-0.027128	0.016775	0.042438	0.101589	...	-0.009118	NaN	0.018422	0.148280	-0.025100	0.0098
EmployeeCount	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	N
EmployeeNumber	-0.010145	-0.050990	0.032916	0.042070	NaN	1.000000	0.017621	0.035179	-0.006888	-0.018519	...	-0.069861	NaN	0.062227	-0.014365	0.023603	0.0103
EnvironmentSatisfaction	0.010146	0.018355	-0.016075	-0.027128	NaN	0.017621	1.000000	-0.049857	-0.008278	0.001212	...	0.007665	NaN	0.003432	-0.002693	-0.019359	0.0276
HourlyRate	0.024287	0.023381	0.031131	0.016775	NaN	0.035179	-0.049857	1.000000	0.042861	-0.027853	...	0.001330	NaN	0.050263	-0.002334	-0.008548	-0.0046
JobInvolvement	0.029820	0.046135	0.008783	0.042438	NaN	-0.006888	-0.008278	0.042861	1.000000	-0.012630	...	0.034297	NaN	0.021523	-0.005533	-0.015338	-0.0146
JobLevel	0.509604	0.002966	0.005303	0.101589	NaN	-0.018519	0.001212	-0.027853	-0.012630	1.000000	...	0.021642	NaN	0.013984	0.782208	-0.018191	0.0378
JobSatisfaction	-0.004892	0.030571	-0.003669	-0.011296	NaN	-0.046247	-0.006784	-0.071335	-0.021476	-0.001944	...	-0.012454	NaN	0.010690	-0.020185	-0.005779	-0.0194
MonthlyIncome	0.497855	0.007707	-0.017014	0.094961	NaN	-0.014829	-0.006259	-0.015794	-0.015271	0.950300	...	0.025873	NaN	0.005408	0.772893	-0.021736	0.0306
MonthlyRate	0.028051	-0.032182	0.027473	-0.026084	NaN	0.012648	0.037600	-0.015297	-0.016322	0.039563	...	-0.004085	NaN	-0.034323	0.026442	0.001467	0.0079
NumCompaniesWorked	0.299635	0.038153	-0.029251	0.126317	NaN	-0.001251	0.012594	0.022157	0.015012	0.142501	...	0.052733	NaN	0.030075	0.237639	-0.066054	-0.0083
PercentSalaryHike	0.003634	0.022704	0.040235	-0.011111	NaN	-0.012944	-0.031701	-0.009062	-0.017205	-0.034730	...	-0.040490	NaN	0.007528	-0.020608	-0.005221	-0.0032
PerformanceRating	0.001904	0.000473	0.027110	-0.024539	NaN	-0.020359	-0.029548	-0.002172	-0.029071	-0.021222	...	-0.031351	NaN	0.003506	0.006744	-0.015579	0.0025
RelationshipSatisfaction	0.053535	0.007846	0.006557	-0.009118	NaN	-0.069861	0.007665	0.001330	0.034297	0.021642	...	1.000000	NaN	-0.045952	0.024054	0.002497	0.0196
StandardHours	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	N
StockOptionLevel	0.037510	0.042143	0.044872	0.018422	NaN	0.062227	0.003432	0.050263	0.021523	0.013984	...	-0.045952	NaN	1.000000	0.010136	0.011274	0.0041
TotalWorkingYears	0.680381	0.014515	0.004628	0.148280	NaN	-0.014365	-0.002693	-0.002334	-0.005533	0.782208	...	0.024054	NaN	0.010136	1.000000	-0.035662	0.0010
TrainingTimesLastYear	-0.019621	0.002453	-0.036942	-0.025100	NaN	0.023603	-0.019359	-0.008548	-0.015338	-0.018191	...	0.002497	NaN	0.011274	-0.035662	1.000000	0.0280
WorkLifeBalance	-0.021490	-0.037848	-0.026556	0.009819	NaN	0.010309	0.027627	-0.004607	-0.014617	0.037818	...	0.019604	NaN	0.004129	0.001008	0.028072	1.0000
YearsAtCompany	0.311309	-0.034055	0.009508	0.069114	NaN	-0.011240	0.001458	-0.019582	-0.021355	0.534739	...	0.019367	NaN	0.015058	0.628133	0.003569	0.0120
YearsInCurrentRole	0.212901	0.009932	0.018845	0.060236	NaN	-0.008416	0.018007	-0.024106	0.008717	0.389447	...	-0.015123	NaN	0.050818	0.460365	-0.005738	0.0498
YearsSinceLastPromotion	0.216513	-0.033229	0.010029	0.054254	NaN	-0.009019	0.016194	-0.026716	-0.024184	0.353885	...	0.033493	NaN	0.014352	0.404858	-0.002067	0.0089
YearsWithCurrManager	0.202089	-0.026363	0.014406	0.069065	NaN	-0.009197	-0.004999	-0.020123	0.025976	0.375281	...	-0.000867	NaN	0.024698	0.459188	-0.004096	0.0027

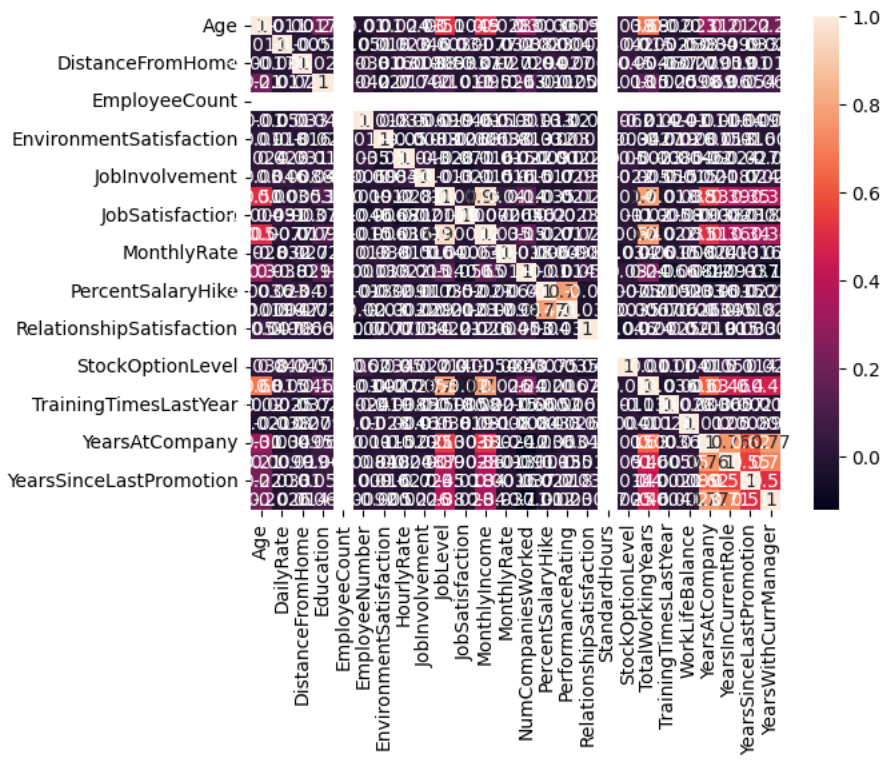
26 rows × 26 columns

```
df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	YearsInCurrent
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	80	0	8	0	1	6	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	80	1	10	3	3	10	

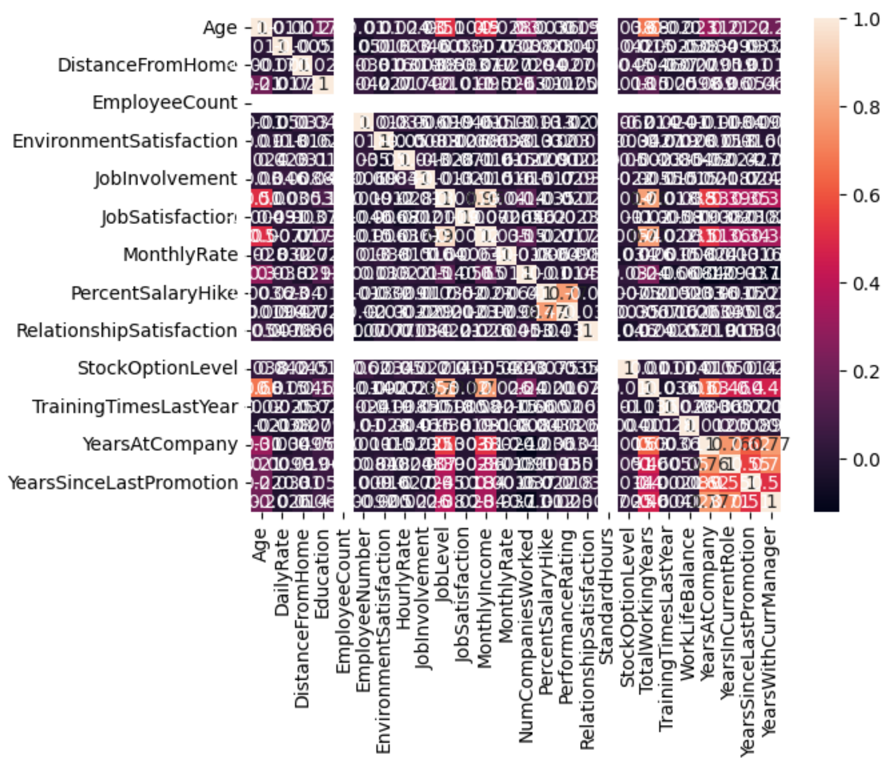
```
numeric_columns = df.select_dtypes(include='number').columns
sns.heatmap(df[numeric_columns].corr(), annot=True)
```

```
<Axes: >
```



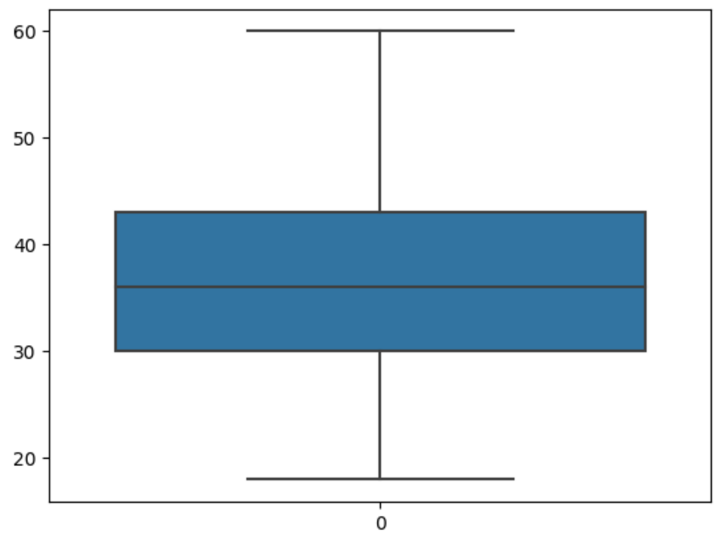

```
sns.heatmap(df.corr(numeric_only=True), annot=True)
```

```
<Axes: >
```



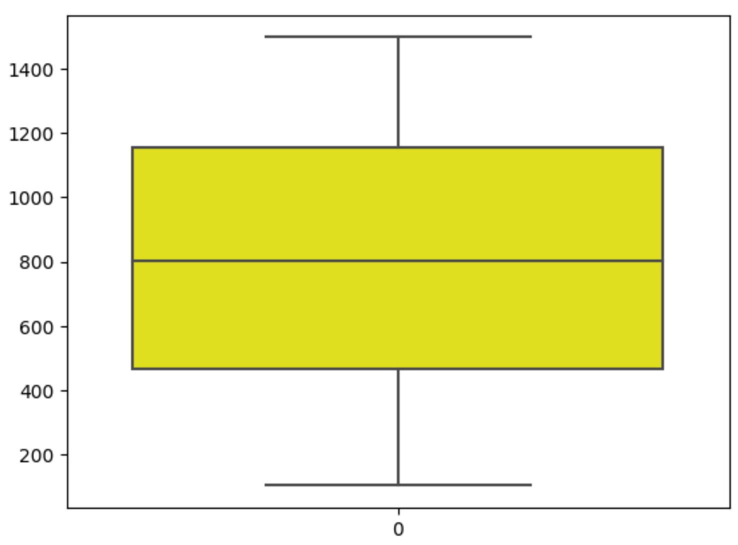

```
sns.boxplot(df['Age'])
```

```
<Axes: >
```




```
sns.boxplot(df['DailyRate'],color='yellow')
```

```
<Axes: >
```




```
#Splitting Dependent and Independent variables
x=df.iloc[:,1:4]
x.head()
```

	Attrition	BusinessTravel	DailyRate
0	Yes	Travel_Rarely	1102
1	No	Travel_Frequently	279
2	Yes	Travel_Rarely	1373
3	No	Travel_Frequently	1392
4	No	Travel_Rarely	591

```
x=df.iloc[:,[i for i in range(df.shape[1]) if i!=1]]
x.head()
```

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany
0	41	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	2	...	1	80	0	8	0	1	6
1	49	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	3	...	4	80	1	10	3	3	10
2	37	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	4	...	2	80	0	7	3	3	0
3	33	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	4	...	3	80	0	8	3	3	8
4	27	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	1	...	4	80	1	6	3	3	2

5 rows × 34 columns

```
y=df.iloc[:,1:2]
y.head()
```

	Attrition
0	Yes
1	No
2	Yes
3	No

```
##LabelEncoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
a=['Age','Department','EducationField','BusinessTravel','Gender','JobRole','MaritalStatus','Over18','OverTime']
for i in a:
  x[i]=le.fit_transform(x[i])
x[a]
```

```
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
<ipython-input-25-c1e99d50f975>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
<ipython-input-25-c1e99d50f975>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
<ipython-input-25-c1e99d50f975>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
<ipython-input-25-c1e99d50f975>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
<ipython-input-25-c1e99d50f975>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  x[i]=le.fit_transform(x[i])
```

	Age	Department	EducationField	BusinessTravel	Gender	JobRole	MaritalStatus	Over18	OverTime
0	23	2	1	2	0	7	2	0	1
1	31	1	1	1	1	6	1	0	0
2	19	1	4	2	1	2	2	0	1
3	15	1	1	1	0	6	1	0	1
4	9	1	3	2	1	2	1	0	0
...	...	...	...	...	...	...	...	...	...
1465	18	1	3	1	1	2	1	0	0
1466	21	1	3	2	1	0	1	0	0
1467	9	1	1	2	1	4	1	0	1
1468	31	2	3	1	1	7	1	0	0
1469	16	1	3	2	1	2	1	0	0

1470 rows × 9 columns

```
y['Attrition']=le.fit_transform(y['Attrition'])
```

```
y.head()
```

	Attrition
0	1
1	0
2	1
3	0
4	0

```
#feature scaling
from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
x_scaled=pd.DataFrame(ms.fit_transform(x),columns=x.columns)
```

```
x_scaled
```

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCo
0	0.547619	1.0	0.715820	1.0	0.000000	0.25	0.2	0.0	0.000000	0.333333	...	0.000000	0.0	0.000000	0.200	0.000000	0.000000	
1	0.738095	0.5	0.126700	0.5	0.250000	0.00	0.2	0.0	0.000484	0.666667	...	1.000000	0.0	0.333333	0.250	0.500000	0.666667	
2	0.452381	1.0	0.909807	0.5	0.035714	0.25	0.8	0.0	0.001451	1.000000	...	0.333333	0.0	0.000000	0.175	0.500000	0.666667	
3	0.357143	0.5	0.923407	0.5	0.071429	0.75	0.2	0.0	0.001935	1.000000	...	0.666667	0.0	0.000000	0.200	0.500000	0.666667	
4	0.214286	1.0	0.350036	0.5	0.035714	0.00	0.6	0.0	0.002903	0.000000	...	1.000000	0.0	0.333333	0.150	0.500000	0.666667	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1465	0.428571	0.5	0.559771	0.5	0.785714	0.25	0.6	0.0	0.996613	0.666667	...	0.666667	0.0	0.333333	0.425	0.500000	0.666667	
1466	0.500000	1.0	0.365784	0.5	0.178571	0.00	0.6	0.0	0.997097	1.000000	...	0.000000	0.0	0.333333	0.225	0.833333	0.666667	
1467	0.214286	1.0	0.037938	0.5	0.107143	0.50	0.2	0.0	0.998065	0.333333	...	0.333333	0.0	0.333333	0.150	0.000000	0.666667	
1468	0.738095	0.5	0.659270	1.0	0.035714	0.50	0.6	0.0	0.998549	1.000000	...	1.000000	0.0	0.000000	0.425	0.500000	0.333333	
1469	0.380952	1.0	0.376521	0.5	0.250000	0.50	0.6	0.0	1.000000	0.333333	...	0.000000	0.0	0.000000	0.150	0.500000	1.000000	

1470 rows × 34 columns

```
#Splitting data into train and test data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.3)
```

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

```
((1029, 34), (441, 34), (1029, 1), (441, 1))
```

```
x_train.head()
```

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCo
694	0.357143	1.0	0.692198	0.5	0.000000	0.50	0.2	0.0	0.468312	0.333333	...	0.000000	0.0	0.000000	0.150	0.500000	0.666667	
1057	0.261905	0.5	0.009306	1.0	0.428571	0.50	1.0	0.0	0.718916	0.000000	...	0.000000	0.0	0.000000	0.175	0.666667	0.000000	
1156	0.523810	1.0	0.559771	0.5	0.500000	0.50	0.2	0.0	0.787131	0.000000	...	1.000000	0.0	0.666667	0.450	0.333333	0.666667	
798	0.357143	1.0	0.654975	0.5	0.857143	0.50	0.6	0.0	0.535559	0.000000	...	0.333333	0.0	0.000000	0.125	0.000000	0.666667	
1451	0.476190	1.0	0.173944	1.0	0.321429	0.25	0.2	0.0	0.986938	0.000000	...	0.666667	0.0	0.333333	0.250	0.166667	0.666667	

5 rows × 34 columns

• Model Building

```
o    Import the model building Libraries
o    Initializing the model
o    Training and testing the model
```

- Evaluation of Model
- Save the Model

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

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

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
▾ LogisticRegression
LogisticRegression()
```

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

```
pred
```

```
array([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, 1, 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,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 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, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 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, 1, 0, 0, 0, 0, 1, 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, 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,
       1, 0, 0, 0, 1, 0, 0, 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, 0, 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, 1, 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, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0])
```

```
y_test
```

	Attrition
1415	0
1073	0
54	0
348	0
953	1
...	...
320	0
183	0
1219	0
1283	0
1164	0

441 rows × 1 columns

```
#Evaluation of Model
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_auc_score,roc_curve
accuracy_score(y_test,pred)
```

```
0.8639455782312925
```

```
confusion_matrix(y_test,pred)
```

```
array([[360,  13],
       [ 47,  21]])
```

```
print(y_test.shape)
print(pred.shape)
```

```
(441, 1)
(441,)
```

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

```
              precision    recall  f1-score   support

           0       0.88      0.97      0.92       373
           1       0.62      0.31      0.41        68

    accuracy                           0.86       441
   macro avg       0.75      0.64      0.67       441
weighted avg       0.84      0.86      0.84       441
```

Random Forest and Decision Tree

```
#Accuracy score
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_auc_score,roc_curve
```

```
accuracy_score(y_test,pred)
```

```
0.8639455782312925
```

```
from sklearn.tree import DecisionTreeClassifier
model1 = DecisionTreeClassifier(max_depth=4,splitter='best',criterion='entropy')
model1.fit(x_train,y_train)
d_y_predict = model1.predict(x_test)
```

```
d_y_predict
```

```
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 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, 1, 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, 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, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 1, 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, 1, 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, 1, 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, 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, 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])
```

```
#Evaluating Metrics- Decision Tree
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print('Testing Accuracy = ', accuracy_score(y_test,d_y_predict))
```

```
Testing Accuracy =  0.8344671201814059
```

```
confusion_matrix(y_test,d_y_predict)
```

```
array([[359,  14],
       [ 59,   9]])
```

```
print(classification_report(y_test,d_y_predict))
```

```
              precision    recall  f1-score   support

           0       0.86      0.96      0.91       373
           1       0.39      0.13      0.20        68

    accuracy                           0.83       441
   macro avg       0.63      0.55      0.55       441
weighted avg       0.79      0.83      0.80       441
```

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

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

```
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']

}
```

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

```
from sklearn.ensemble import RandomForestClassifier
model2 =RandomForestClassifier(criterion='entropy')
model2.fit(x_train,y_train)
```

```
<ipython-input-53-8761b0c09731>:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  model2.fit(x_train,y_train)
▾          RandomForestClassifier
RandomForestClassifier(criterion='entropy')
```

```
r_y_predict = model2.predict(x_test)
r_y_predict_train = model2.predict(x_train)
```

```
print('Testing Accuracy = ', accuracy_score(y_test,r_y_predict))
print('Training Accuracy = ', accuracy_score(y_train,r_y_predict_train))
```

```
Testing Accuracy =  0.8684807256235828
Training Accuracy =  1.0
```

```
print(classification_report(y_test,r_y_predict))
```

```
              precision    recall  f1-score   support

           0       0.87      0.99      0.93       373
           1       0.86      0.18      0.29        68

    accuracy                           0.87       441
   macro avg       0.86      0.59      0.61       441
weighted avg       0.87      0.87      0.83       441
```

```
confusion_matrix(y_test,r_y_predict)
```

```
array([[371,   2],
       [ 56,  12]])
```