Importing necessary libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing dataset Employ Atrrition

In [2]:
```
data=pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

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

Out[3]:

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

5 rows × 35 columns

In [4]:
```
data.tail()
```

Out[4]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	Emp
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical	
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical	
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical	
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical	

5 rows × 35 columns

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

In [6]:

```
data.describe()
```

Out[6]:

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

8 rows × 26 columns

Checking for null values

In [7]:
```
data.isnull().any()
```

Out[7]:
```
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
```

We observe that there are no null values in the dataset

Therefor there is no need to handle null values

Data Visualization

In [8]:
```
corr=data.corr()
corr
```

Out[8]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber
Age	1.000000	0.010661	-0.001686	0.208034	NaN	-0.010145
DailyRate	0.010661	1.000000	-0.004985	-0.016806	NaN	-0.050990
DistanceFromHome	-0.001686	-0.004985	1.000000	0.021042	NaN	0.032916
Education	0.208034	-0.016806	0.021042	1.000000	NaN	0.042070
EmployeeCount	NaN	NaN	NaN	NaN	NaN	NaN
EmployeeNumber	-0.010145	-0.050990	0.032916	0.042070	NaN	1.000000

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber
EnvironmentSatisfaction	0.010146	0.018355	-0.016075	-0.027128	NaN	0.017621
HourlyRate	0.024287	0.023381	0.031131	0.016775	NaN	0.035179
JobInvolvement	0.029820	0.046135	0.008783	0.042438	NaN	-0.006888
JobLevel	0.509604	0.002966	0.005303	0.101589	NaN	-0.018519
JobSatisfaction	-0.004892	0.030571	-0.003669	-0.011296	NaN	-0.046247
MonthlyIncome	0.497855	0.007707	-0.017014	0.094961	NaN	-0.014829
MonthlyRate	0.028051	-0.032182	0.027473	-0.026084	NaN	0.012648
NumCompaniesWorked	0.299635	0.038153	-0.029251	0.126317	NaN	-0.001251
PercentSalaryHike	0.003634	0.022704	0.040235	-0.011111	NaN	-0.012944
PerformanceRating	0.001904	0.000473	0.027110	-0.024539	NaN	-0.020359
RelationshipSatisfaction	0.053535	0.007846	0.006557	-0.009118	NaN	-0.069861
StandardHours	NaN	NaN	NaN	NaN	NaN	NaN
StockOptionLevel	0.037510	0.042143	0.044872	0.018422	NaN	0.062227
TotalWorkingYears	0.680381	0.014515	0.004628	0.148280	NaN	-0.014365
TrainingTimesLastYear	-0.019621	0.002453	-0.036942	-0.025100	NaN	0.023603
WorkLifeBalance	-0.021490	-0.037848	-0.026556	0.009819	NaN	0.010309
YearsAtCompany	0.311309	-0.034055	0.009508	0.069114	NaN	-0.011240
YearsInCurrentRole	0.212901	0.009932	0.018845	0.060236	NaN	-0.008416
YearsSinceLastPromotion	0.216513	-0.033229	0.010029	0.054254	NaN	-0.009019
YearsWithCurrManager	0.202089	-0.026363	0.014406	0.069065	NaN	-0.009197

26 rows × 26 columns

In [9]:
```
plt.subplots(figsize=(20,10))
sns.heatmap(corr,annot=True)
```

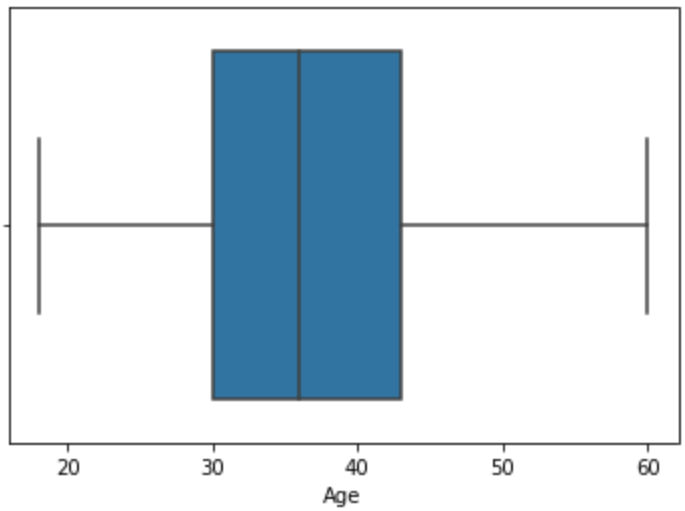
Out[9]:
```
<AxesSubplot:>
```

Outlier detection

In [10]:
```
sns.boxplot(data.Age)
```

```
C:\Users\ishan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
the following variable as a keyword arg: x. From version 0.12, the only valid positional a
rgument will be `data`, and passing other arguments without an explicit keyword will resul
t in an error or misinterpretation.
  warnings.warn(
```

Out[10]:
```
<AxesSubplot:xlabel='Age'>
```



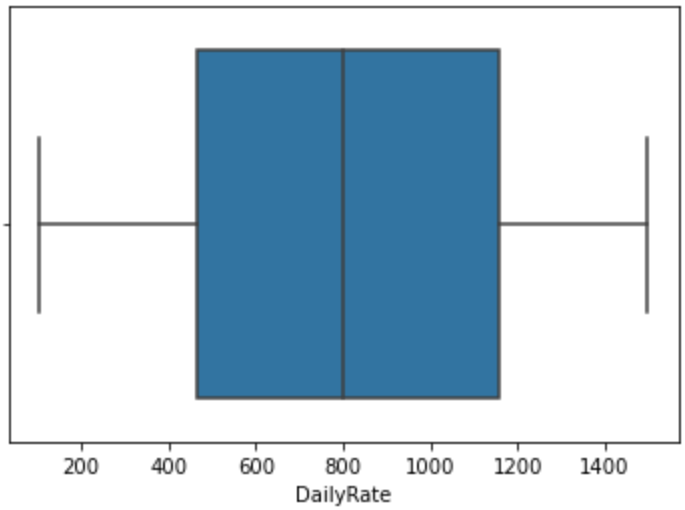
In [11]:
```
sns.boxplot(data.DailyRate)
```

```
C:\Users\ishan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
the following variable as a keyword arg: x. From version 0.12, the only valid positional a
rgument will be `data`, and passing other arguments without an explicit keyword will resul
```

```
t in an error or misinterpretation.
  warnings.warn(
```

Out[11]:

```
<AxesSubplot:xlabel='DailyRate'>
```



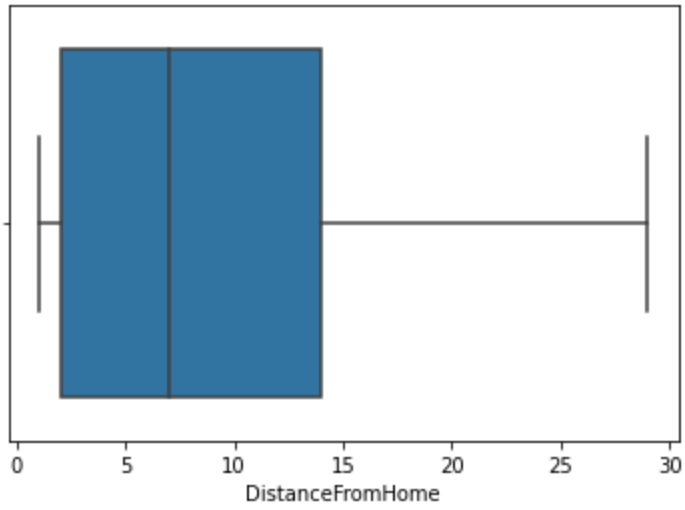
In [12]:

```
sns.boxplot(data.DistanceFromHome)
```

```
C:\Users\ishan\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass
the following variable as a keyword arg: x. From version 0.12, the only valid positional a
rgument will be `data`, and passing other arguments without an explicit keyword will resul
t in an error or misinterpretation.
  warnings.warn(
```

Out[12]:

```
<AxesSubplot:xlabel='DistanceFromHome'>
```



Splitting Dependent and Independent variables

In [13]:

```
X=data.loc[:, data.columns != 'Attrition']
X
```

Out[13]:

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCou
0	41	Travel_Rarely	1102	Sales	1	2	Life Sciences	
1	49	Travel_Frequently	279	Research & Development	8	1	Life Sciences	
2	37	Travel_Rarely	1373	Research & Development	2	2	Other	

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCou
3	33	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	
4	27	Travel_Rarely	591	Research & Development	2	1	Medical	
...	...	...	...	...	...	...	...	
1465	36	Travel_Frequently	884	Research & Development	23	2	Medical	
1466	39	Travel_Rarely	613	Research & Development	6	1	Medical	
1467	27	Travel_Rarely	155	Research & Development	4	3	Life Sciences	
1468	49	Travel_Frequently	1023	Sales	2	3	Medical	
1469	34	Travel_Rarely	628	Research & Development	8	3	Medical	

1470 rows × 34 columns

In [14]:
```
Y=data.loc[:,data.columns=='Attrition']
Y
```

Out[14]:

	Attrition
0	Yes
1	No
2	Yes
3	No
4	No
...	...
1465	No
1466	No
1467	No
1468	No
1469	No

1470 rows × 1 columns

Encoding

One hot encoding

In [15]:
```
print(data['JobRole'].unique())
```

```
['Sales Executive' 'Research Scientist' 'Laboratory Technician'
 'Manufacturing Director' 'Healthcare Representative' 'Manager'
 'Sales Representative' 'Research Director' 'Human Resources']
```

In [16]:
```
ver18'].unique())
```

```
['Y']
```

In [17]:
```
print(data['OverTime'].unique())
```

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

In [18]:
```
print(data['MaritalStatus'].unique())
```

```
['Single' 'Married' 'Divorced']
```

In [19]:
```
print(data['Gender'].unique())
```

```
['Female' 'Male']
```

In [20]:
```
print(data['EducationField'].unique())
```

```
['Life Sciences' 'Other' 'Medical' 'Marketing' 'Technical Degree'
 'Human Resources']
```

In [21]:
```
print(data['BusinessTravel'].unique())
```

```
['Travel_Rarely' 'Travel_Frequently' 'Non-Travel']
```

In [22]:
```
print(data['Department'].unique())
```

```
['Sales' 'Research & Development' 'Human Resources']
```

In [23]:
```
data['Gender'].value_counts()
```

Out[23]:
```
Male      882
Female    588
Name: Gender, dtype: int64
```

In [24]:
```
data['EducationField'].value_counts()
```

Out[24]:
```
Life Sciences       606
Medical             464
Marketing           159
Technical Degree    132
Other                82
Human Resources      27
Name: EducationField, dtype: int64
```

In [25]:
```
data['Over18'].value_counts()
```

Out[25]:
```
Y    1470
Name: Over18, dtype: int64
```

In [26]:
```
data['OverTime'].value_counts()
```

Out[26]:
```
No     1054
Yes     416
Name: OverTime, dtype: int64
```

In [27]:
```
data['MaritalStatus'].value_counts()
```

Out[27]:
```
Married     673
Single      470
Divorced    327
Name: MaritalStatus, dtype: int64
```

In [28]:
```
data['JobRole'].value_counts()
```

Out[28]:
```
Sales Executive              326
Research Scientist           292
Laboratory Technician        259
Manufacturing Director       145
Healthcare Representative    131
Manager                      102
Sales Representative          83
Research Director             80
Human Resources               52
Name: JobRole, dtype: int64
```

In [29]:
```
data['BusinessTravel'].value_counts()
```

Out[29]:
```
Travel_Rarely        1043
Travel_Frequently     277
Non-Travel            150
Name: BusinessTravel, dtype: int64
```

In [30]:
```
data['Department'].value_counts()
```

Out[30]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

In [31]:
```
X = pd.get_dummies(X, columns = ['MaritalStatus','JobRole','Gender','Department','Educatio
print(X)
```

```
      Age  DailyRate  DistanceFromHome  Education  EmployeeCount  \
0      41       1102                 1          2              1
1      49        279                 8          1              1
2      37       1373                 2          2              1
3      33       1392                 3          4              1
4      27        591                 2          1              1
...   ...        ...               ...        ...            ...
1465   36        884                23          2              1
1466   39        613                 6          1              1
1467   27        155                 4          3              1
1468   49       1023                 2          3              1
1469   34        628                 8          3              1

      EmployeeNumber  EnvironmentSatisfaction  HourlyRate  JobInvolvement  \
0                  1                        2          94               3
1                  2                        3          61               2
2                  4                        4          92               2
3                  5                        4          56               3
4                  7                        1          40               3
...              ...                      ...         ...             ...
1465            2061                        3          41               4
1466            2062                        4          42               2
1467            2064                        2          87               4
1468            2065                        4          63               2
1469            2068                        2          82               4

                                                   ...  EducationField_Marketing  \
```



```
  EducationField_Medical  \
```

```
0            2  ...                       0                       0
1            2  ...                       0                       0
2            1  ...                       0                       0
3            1  ...                       0                       0
4            1  ...                       0                       1
...        ...  ...                     ...                     ...
1465         2  ...                       0                       1
1466         3  ...                       0                       1
1467         2  ...                       0                       0
1468         2  ...                       0                       1
1469         2  ...                       0                       1

      EducationField_Other  EducationField_Technical Degree  \
0                        0                                 0
1                        0                                 0
2                        1                                 0
3                        0                                 0
4                        0                                 0
...                    ...                               ...
1465                     0                                 0
1466                     0                                 0
1467                     0                                 0
1468                     0                                 0
1469                     0                                 0

      BusinessTravel_Non-Travel  BusinessTravel_Travel_Frequently  \
0                             0                                 0
1                             0                                 1
2                             0                                 0
3                             0                                 1
4                             0                                 0
...                         ...                               ...
1465                          0                                 1
1466                          0                                 0
1467                          0                                 0
1468                          0                                 1
1469                          0                                 0

      BusinessTravel_Travel_Rarely  Over18_Y  OverTime_No  OverTime_Yes
0                                1         1            0             1
1                                0         1            1             0
2                                1         1            0             1
3                                0         1            0             1
4                                1         1            1             0
...                            ...       ...          ...           ...
1465                             0         1            1             0
1466                             1         1            1             0
1467                             1         1            0             1
1468                             0         1            1             0
1469                             1         1            1             0

[1470 rows x 55 columns]
```

In [32]:

```
X.head()
```

Out[32]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	Hou
0	41	1102	1	2	1	1	2	
1	49	279	8	1	1	2	3	
2	37	1373	2	2	1	4	4	
3	33	1392	3	4	1	5	4	

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	Hou
4	27	591	2	1	1	7	1	

5 rows × 55 columns

Splitting into training and testing dataset

In [33]:
```
from sklearn.model_selection import train_test_split
```

In [34]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=0)
X_train.shape,Y_train.shape,X_test.shape,Y_test.shape
```

Out[34]:
```
((1176, 55), (1176, 1), (294, 55), (294, 1))
```

Feature Scaling

In [35]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

In [36]:
```
X_train=sc.fit_transform(X_train)
X_test=sc.fit_transform(X_test)
```

In [37]:
```
X_train
```

Out[37]:
```
array([[ 2.3389367 , -0.48557354,  1.45567735, ...,  0.        ,
        -1.62507442,  1.62507442],
       [ 0.9043263 ,  0.36465323,  2.31628752, ...,  0.        ,
         0.61535643, -0.61535643],
       [ 0.35255307, -1.23722329,  2.0703989 , ...,  0.        ,
         0.61535643, -0.61535643],
       ...,
       [ 0.68361701,  0.92900666, -0.88026453, ...,  0.        ,
        -1.62507442,  1.62507442],
       [ 0.13184377, -1.31608491, -0.88026453, ...,  0.        ,
        -1.62507442,  1.62507442],
       [ 0.35255307, -0.35495899,  0.10328995, ...,  0.        ,
         0.61535643, -0.61535643]])
```

Model Building

Import the model building Libraries 1.Initializing the model 2.Training and testing the model 3.Evaluation of Model 4.Save the Model

Logistic Regression

In [38]:
```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
```

In [39]:
```
model.fit(X_train,Y_train)
```

```
C:\Users\ishan\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: 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().
  return f(*args, **kwargs)
```

Out[39]:
```
LogisticRegression()
```

In [40]:
```
pred=model.predict(X_test)
pred
```

Out[40]:
```
array(['No', 'No', 'Yes', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'Yes', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'Yes', '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', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'Yes', 'No',
       'No', 'No', 'Yes', 'Yes', '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',
       'Yes', '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', 'Yes', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', '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', 'Yes', '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', 'Yes', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No'], dtype=object)
```

In [41]:
```
Y_test
```

Out[41]:

	Attrition
442	No
1091	No
981	Yes
785	No
1332	Yes
...	...
1439	No
481	No
124	Yes
198	No
1229	No

294 rows × 1 columns

In [61]:
```
#Evaluation of classification model
```

In [42]:
```
#Accuracy score
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_auc_
```

In [43]:
```
accuracy_score(Y_test,pred)
```

Out[43]:
```
0.8809523809523809
```

In [44]:
```
confusion_matrix(Y_test,pred)
```

Out[44]:
```
array([[238,   7],
       [ 28,  21]], dtype=int64)
```

In [45]:
```
print(classification_report(Y_test,pred))
```

```
              precision    recall  f1-score   support

          No       0.89      0.97      0.93       245
         Yes       0.75      0.43      0.55        49

    accuracy                           0.88       294
   macro avg       0.82      0.70      0.74       294
weighted avg       0.87      0.88      0.87       294
```

Decision tree

In [46]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [47]:
```
dtc_cv=DecisionTreeClassifier(criterion= 'entropy',
 max_depth=3,
 max_features='sqrt',
 splitter='best')
dtc_cv.fit(X_train,Y_train)
```

Out[47]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=3, max_features='sqrt')
```

In [48]:
```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,Y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

In [49]:
```
y_pred
```

Out[49]:
```
array(['No', 'No', 'No', 'No', 'No', 'Yes', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'Yes', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'Yes', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes',
       'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'Yes',
       'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'Yes', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'Yes', 'Yes', 'Yes', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No',
       'No', 'Yes', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No',
       'Yes', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'Yes', 'Yes', 'No', 'No', 'No', 'No', 'Yes',
       'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'Yes', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No',
       'No', 'Yes', 'Yes', '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', 'Yes',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes',
       'No', 'No', 'Yes', 'No', 'No', 'Yes', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No'], dtype=object)
```

In [50]:
```
#Evaluating the model
print("Accuracy:",accuracy_score(Y_test, y_pred))
```

```
Accuracy: 0.7925170068027211
```

In [51]:
```
print(classification_report(Y_test,y_pred))
```

```
              precision    recall  f1-score   support

          No       0.87      0.88      0.88       245
         Yes       0.38      0.37      0.37        49

    accuracy                           0.79       294
   macro avg       0.62      0.62      0.62       294
weighted avg       0.79      0.79      0.79       294
```

Random Forest

In [56]:
```
from sklearn.ensemble import RandomForestClassifier
classifier= RandomForestClassifier(n_estimators= 10, criterion="entropy")
classifier.fit(X_train, Y_train)
```

```
C:\Users\ishan\AppData\Local\Temp/ipykernel_34008/2248966046.py: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().
  classifier.fit(X_train, Y_train)
```

Out[56]:
```
RandomForestClassifier(criterion='entropy', n_estimators=10)
```

In [57]:
```
Y_Pred=classifier.predict(X_test)
```

In [58]:
```
Y_Pred
```

Out[58]:
```
array(['No', 'No', 'No', 'No', 'Yes', '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', 'Yes', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'Yes', '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', 'Yes',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', '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', 'Yes', 'Yes', 'No',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'Yes',
       'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No', 'No',
       'No', 'No', 'No', 'No', 'No', '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'], dtype=object)
```

In [59]:
```
#Evaluationg the model
print("Accuracy:",accuracy_score(Y_test, Y_Pred))
```

```
Accuracy: 0.8741496598639455
```

In [85]:
```
print(classification_report(Y_test,Y_Pred))
```

```
              precision    recall  f1-score   support

          No       0.87      1.00      0.93       245
         Yes       0.93      0.27      0.41        49

    accuracy                           0.87       294
   macro avg       0.90      0.63      0.67       294
weighted avg       0.88      0.87      0.84       294
```

In [ ]:
```

```