In [1]:

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

In [2]:

```
df=pd.read_csv('Employee-Attrition.csv')
```

In [3]:

```
df.head()
```

Out[3]:

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

5 rows × 35 columns

DATA PRE-PROCESSING

In [4]:

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

Out[4]:

```
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
```

No Null values detected

In [5]:

```
df.shape
```

Out[5]:

```
(1470, 35)
```

In [6]:

```
df.describe()
```

Out[6]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfactio
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.00000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.72176
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.09308
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.00000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.00000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.00000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.00000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.00000

8 rows × 26 columns

In [7]:

```
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
 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
```

DATA VISUALISATION

In [8]:

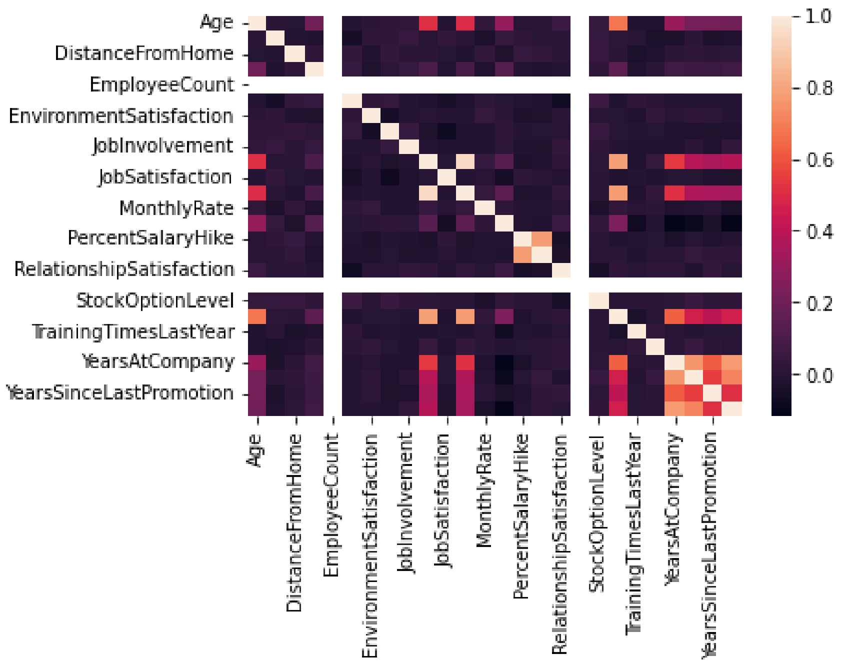
```
corr=df.corr()
```

In [9]:

```
sns.heatmap(corr)
```

Out[9]:

```
<AxesSubplot:>
```



In [10]:

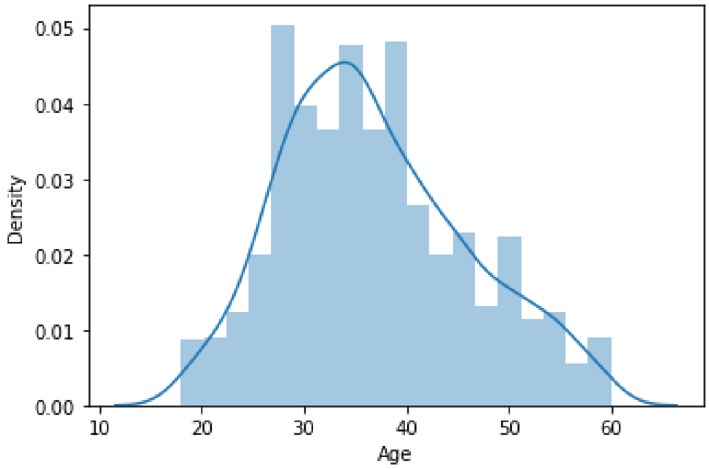
```
sns.distplot(df.Age)
```

```
C:\Users\kommi\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[10]:

```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```

```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```



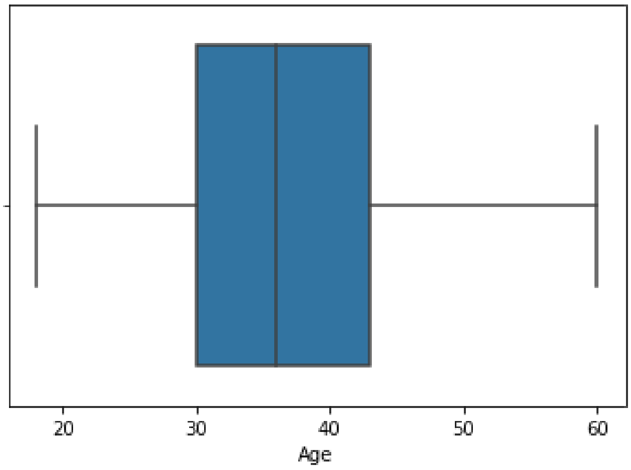
In [11]:

```
sns.boxplot(df.Age)
plt.figure(figsize=(10, 6))
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[11]:

```
<Figure size 720x432 with 0 Axes>
```



```
<Figure size 720x432 with 0 Axes>
```

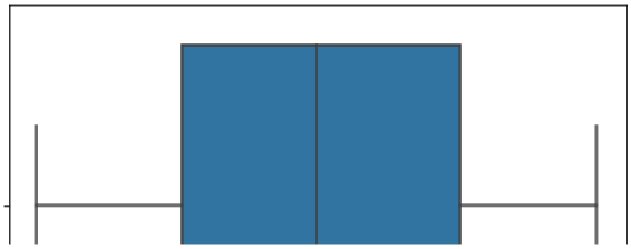
In [12]:

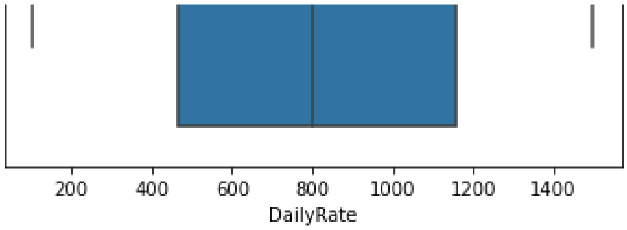
```
sns.boxplot(df.DailyRate)
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[12]:

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





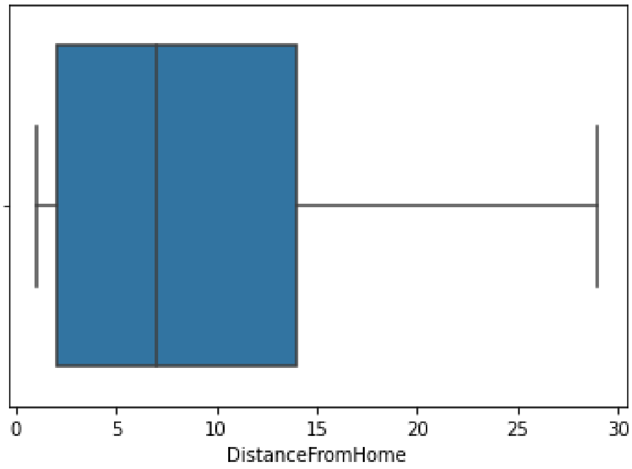
In [13]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[13]:

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



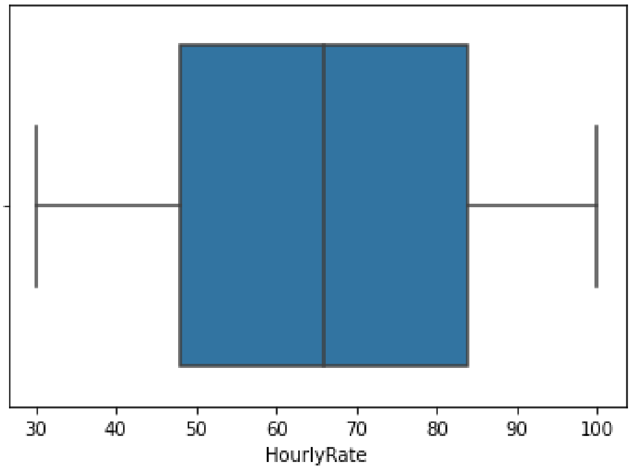
In [14]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[14]:

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



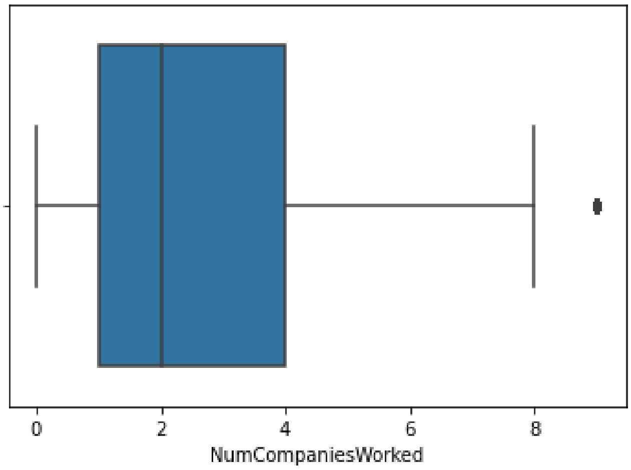
In [15]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[15]:

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



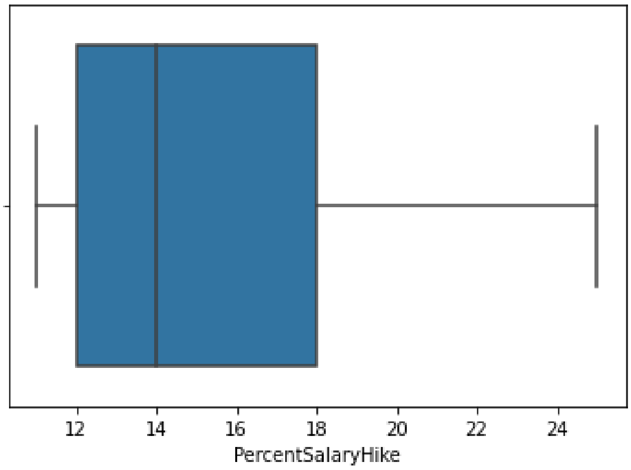
In [16]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[16]:

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



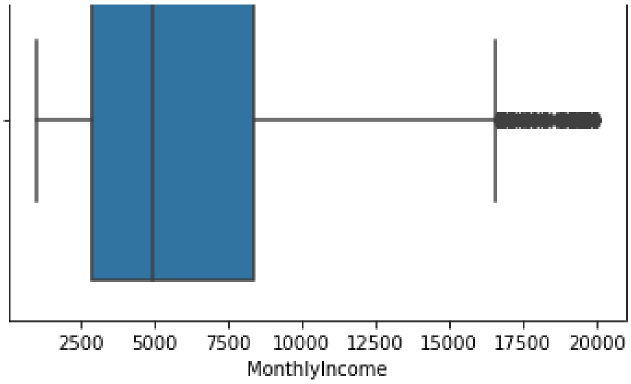
In [17]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[17]:

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



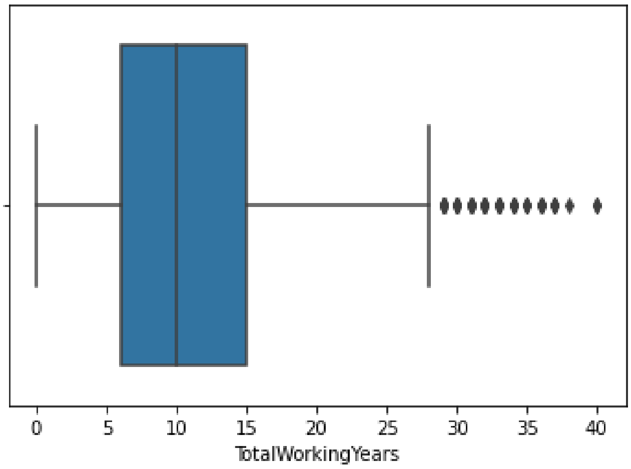
In [18]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[18]:

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



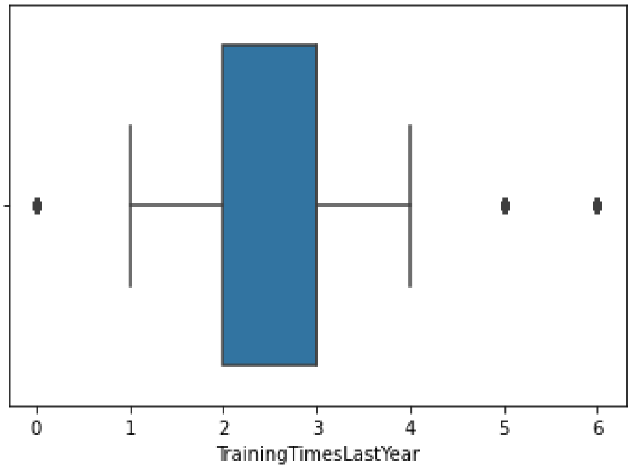
In [19]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[19]:

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



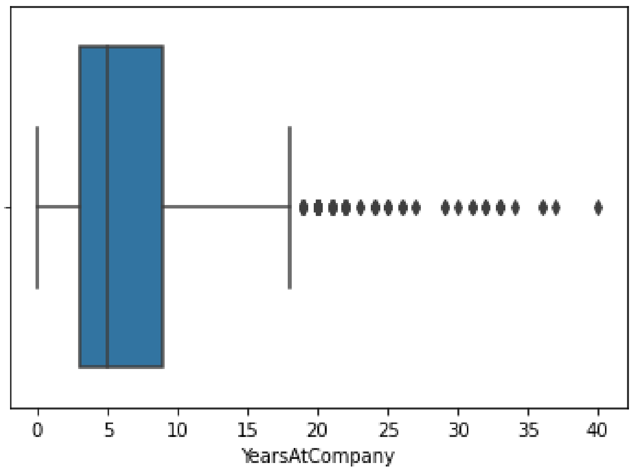
In [20]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[20]:

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



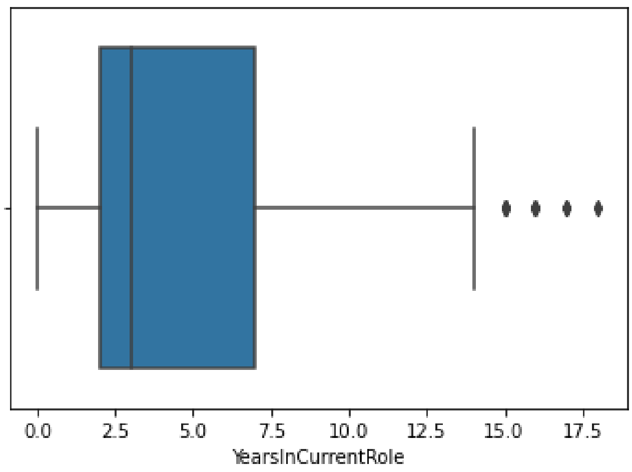
In [21]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[21]:

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



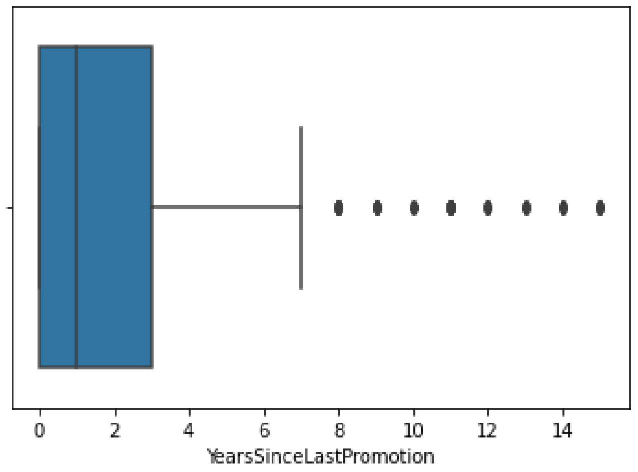
In [22]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[22]:

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



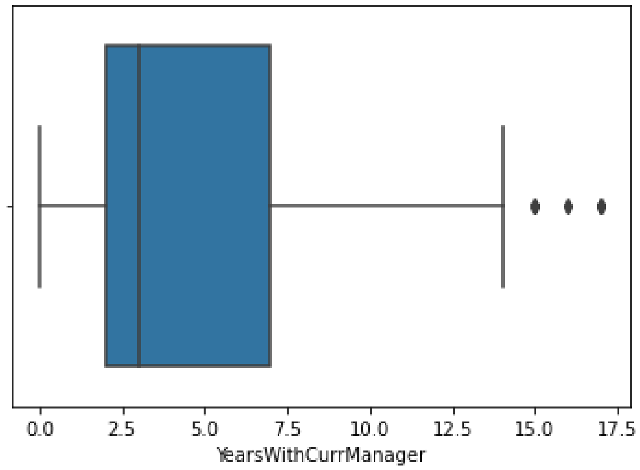
In [23]:

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

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[23]:

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



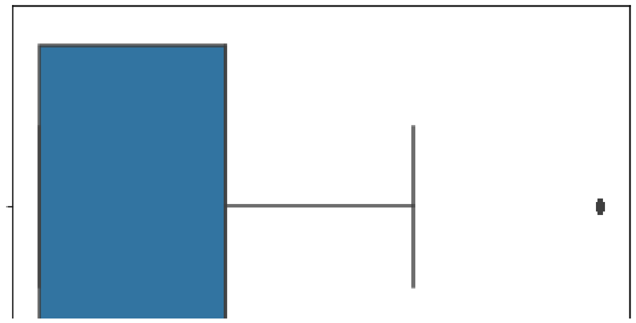
In [24]:

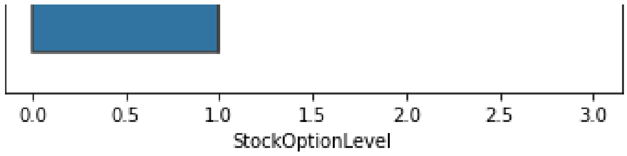
```
sns.boxplot(df.StockOptionLevel)
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[24]:

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





Outliers detected in Following attributes

```
* YearsWithCurrManager
* YearsSinceLastPromotion
* YearsInCurrentRole
* YearsAtCompany
* TrainingTimesLastYear
* TotalWorkingYears
* NumCompaniesWorked
* StockOptionlevel
* Monthly income
```

Now Outlier removal

In [25]:

```
#YearsWithCurrManager
q1=df.YearsWithCurrManager.quantile(0.25)
q3=df.YearsWithCurrManager.quantile(0.75)
IQR=q3-q1
IQR
```

Out[25]:

```
5.0
```

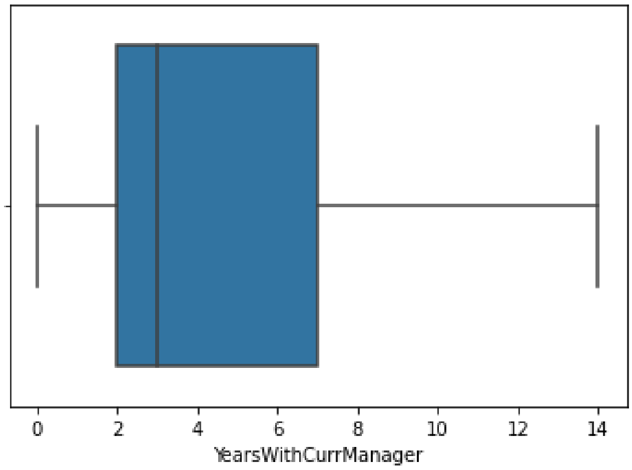
In [26]:

```
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['YearsWithCurrManager']<upper_limit ) ]
sns.boxplot(df['YearsWithCurrManager'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[26]:

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



In [27]:

```
#YearsSinceLastPromotion
```

```
q1=df.YearsSinceLastPromotion.quantile(0.25)
q3=df.YearsSinceLastPromotion.quantile(0.75)
IQR=q3-q1
IQR
```

Out[27]:

```
2.0
```

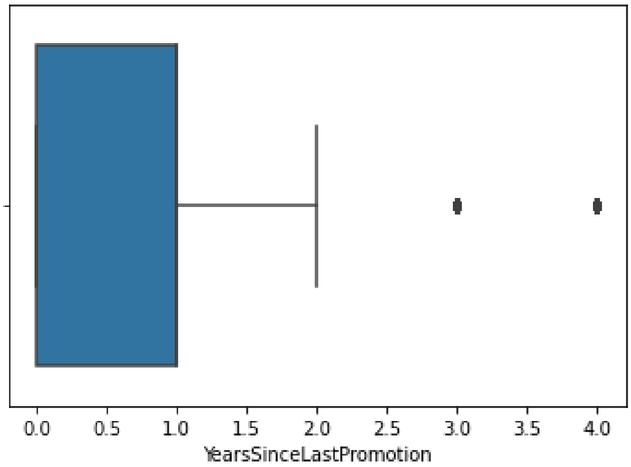
In [28]:

```
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['YearsSinceLastPromotion']<upper_limit ) ]
sns.boxplot(df['YearsSinceLastPromotion'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[28]:

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



In [29]:

```
#YearsWithCurrManager
q1=df.YearsInCurrentRole.quantile(0.25)
q3=df.YearsInCurrentRole.quantile(0.75)
IQR=q3-q1
IQR
```

Out[29]:

```
3.0
```

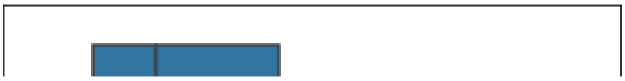
In [30]:

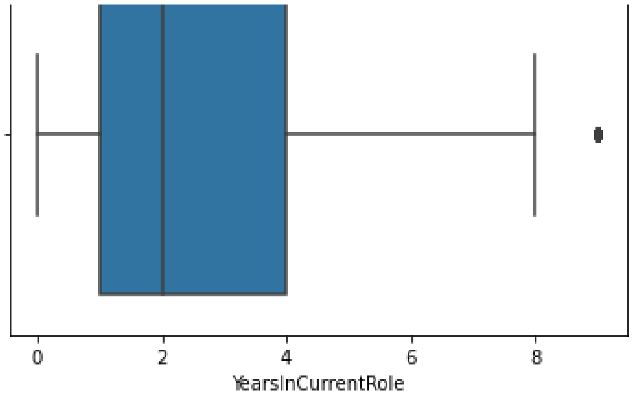
```
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['YearsInCurrentRole']<upper_limit ) ]
sns.boxplot(df['YearsInCurrentRole'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[30]:

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





In [31]:

```
#YearsWithCurrManager
q1=df.YearsAtCompany.quantile(0.25)
q3=df.YearsAtCompany.quantile(0.75)
IQR=q3-q1
IQR
```

Out[31]:

```
5.0
```

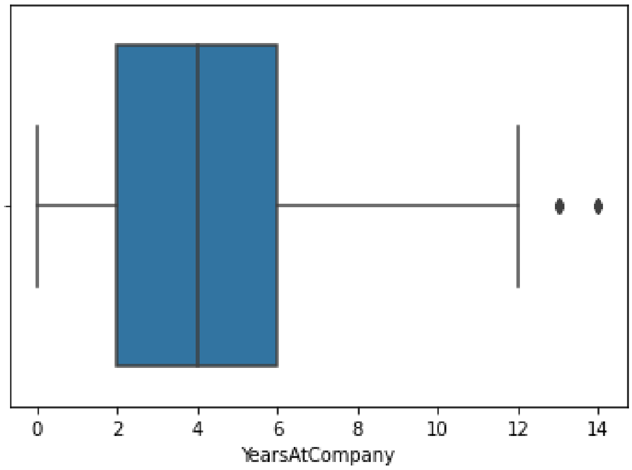
In [32]:

```
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['YearsAtCompany']<upper_limit ) ]
sns.boxplot(df['YearsAtCompany'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[32]:

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



In [33]:

```
#YearsWithCurrManager
q1=df.TrainingTimesLastYear.quantile(0.25)
q3=df.TrainingTimesLastYear.quantile(0.75)
IQR=q3-q1
IQR
```

Out[33]:

```
1.0
```

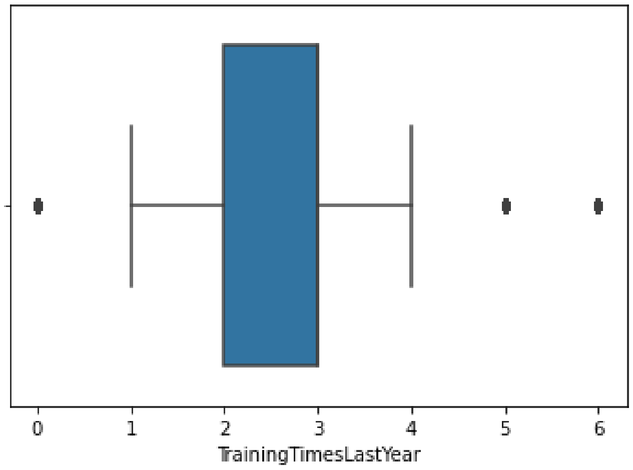
In [34]:

```
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['TrainingTimesLastYear']<upper_limit ) | (df['TrainingTimesLastYear']>lower_l
imit)]
sns.boxplot(df['TrainingTimesLastYear'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[34]:

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



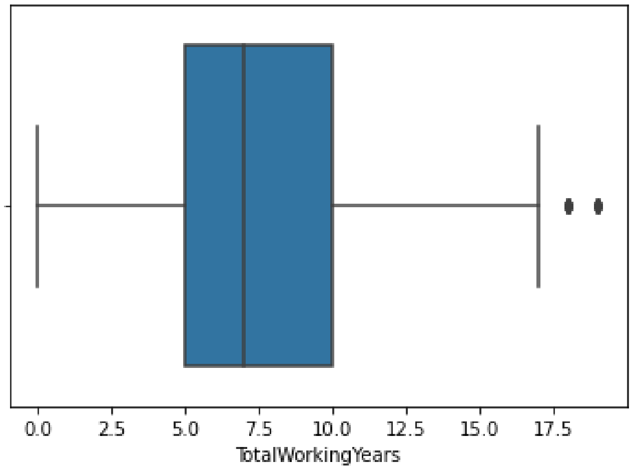
In [35]:

```
#YearsWithCurrManager
q1=df.TotalWorkingYears.quantile(0.25)
q3=df.TotalWorkingYears.quantile(0.75)
IQR=q3-q1
IQR
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['TotalWorkingYears']<upper_limit )]
sns.boxplot(df['TotalWorkingYears'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[35]:

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



In [36]:

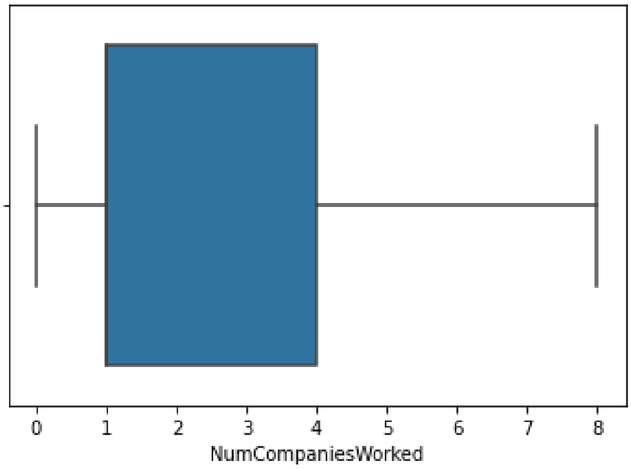
```
#YearsWithCurrManager
```

```
#YearsWithCurrManager
q1=df.NumCompaniesWorked.quantile(0.25)
q3=df.NumCompaniesWorked.quantile(0.75)
IQR=q3-q1
IQR
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[  (df['NumCompaniesWorked']<upper_limit )]
sns.boxplot(df['NumCompaniesWorked'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[36]:

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



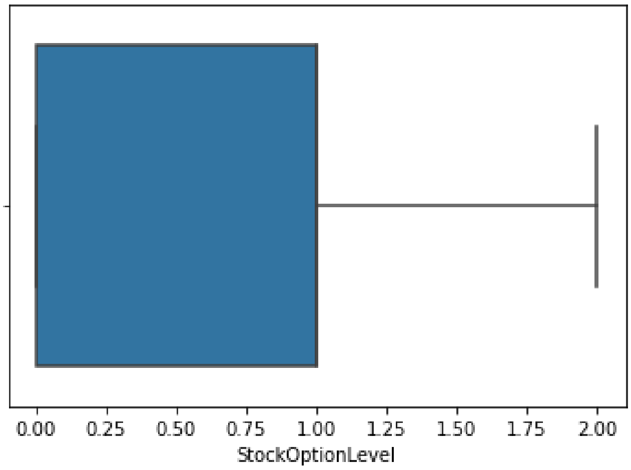
In [37]:

```
#YearsWithCurrManager
q1=df.StockOptionLevel.quantile(0.25)
q3=df.StockOptionLevel.quantile(0.75)
IQR=q3-q1
IQR
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[  (df['StockOptionLevel']<upper_limit )]
sns.boxplot(df['StockOptionLevel'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[37]:

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



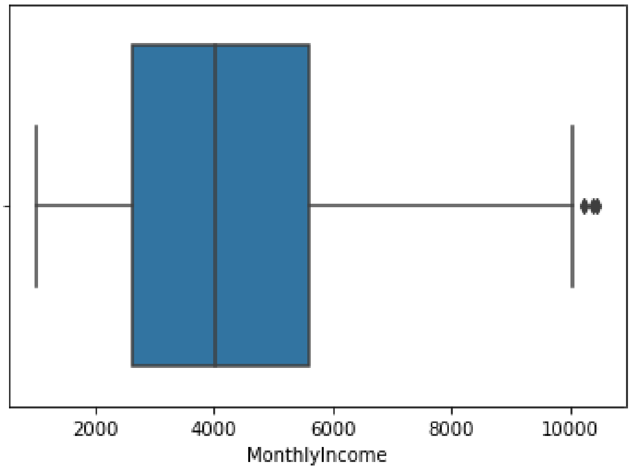
In [38]:

```
#MonthlyIncome
q1=df.MonthlyIncome.quantile(0.25)
q3=df.MonthlyIncome.quantile(0.75)
IQR=q3-q1
IQR
upper_limit=q3+1.5*IQR
lower_limit=q1-1.5*IQR
df=df[ (df['MonthlyIncome']<upper_limit )]
sns.boxplot(df['MonthlyIncome'])
```

```
C:\Users\kommi\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
argument will be `data`, and passing other arguments without an explicit keyword will res
ult in an error or misinterpretation.
  warnings.warn(
```

Out[38]:

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



In [39]:

```
df.describe()
```

Out[39]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction
count	902.000000	902.000000	902.000000	902.000000	902.0	902.000000	902.000000
mean	34.211752	799.273836	9.182927	2.827051	1.0	1027.853659	2.715078
std	8.298172	399.839801	7.925066	1.026302	0.0	613.052180	1.095736
min	18.000000	103.000000	1.000000	1.000000	1.0	1.000000	1.000000
25%	28.000000	465.500000	2.000000	2.000000	1.0	484.250000	2.000000
50%	33.000000	805.000000	7.000000	3.000000	1.0	1010.500000	3.000000
75%	39.000000	1153.000000	14.000000	4.000000	1.0	1576.250000	4.000000
max	60.000000	1498.000000	29.000000	5.000000	1.0	2068.000000	4.000000

8 rows × 26 columns

We can observe many mean values has changed due to removal of outliers

In [40]:

```
df.shape
```

Out[40]:

```
(902, 35)
```

SEPARATING DEPENDENT AND INDEPENDENT VARIABLES

In [41]:

```
x_d=df.iloc[:,:]
X=x_d.drop(columns=["Attrition","EmployeeCount","StandardHours","Over18"],axis=1)
```

In [42]:

```
X.head()
```

Out[42]:

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeNumber	Environ
0	41	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	
1	49	Travel_Frequently	279	Research & Development	8	1	Life Sciences	2	
2	37	Travel_Rarely	1373	Research & Development	2	2	Other	4	
3	33	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	5	
5	32	Travel_Frequently	1005	Research & Development	2	2	Life Sciences	8	

5 rows × 31 columns

In [43]:

```
y=df["Attrition"]
```

In [44]:

```
y.head()
```

Out[44]:

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

In [45]:

```
l=df.iloc[:,19:]
l.head()
```

Out[45]:

	MonthlyRate	NumCompaniesWorked	Over18	OverTime	PercentSalaryHike	PerformanceRating	RelationshipSatisfaction	S
0	19479	8	Y	Yes	11	3	1	
1	24907	1	Y	No	23	4	4	
2	2396	6	Y	Yes	15	3	2	
3	23159	1	Y	Yes	11	3	3	
5	11864	0	Y	No	13	3	3	

In [46]:

```
df.head()
```

Out[46]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1
5	32	No	Travel_Frequently	1005	Research & Development	2	2	Life Sciences	1

5 rows × 35 columns

ENCODING

In [47]:

```
x_encoded=pd.get_dummies(X,columns=["Department","BusinessTravel","EducationField","Gend
er","OverTime","JobRole","MaritalStatus"])
```

In [48]:

```
x_encoded
```

Out[48]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement
0	41	1102	1	2	1	2	94	3
1	49	279	8	1	2	3	61	2
2	37	1373	2	2	4	4	92	2
3	33	1392	3	4	5	4	56	3
5	32	1005	2	2	8	4	79	3
...	...	...	...	...	...	...	...	...
1465	36	884	23	2	2061	3	41	4
1466	39	613	6	1	2062	4	42	2
1467	27	155	4	3	2064	2	87	4
1468	49	1023	2	3	2065	4	63	2
1469	34	628	8	3	2068	2	82	4

902 rows × 50 columns

In [49]:

```
y_encoded=pd.get_dummies(y,drop_first=True)
```

In [50]:

```
y_encoded.head()
```

Out[50]:

	Yes
	Yes
0	1
1	0
2	1
3	0
5	0

FEATURE SCALING

In [51]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [52]:

```
ms=MinMaxScaler()
```

In [53]:

```
X_Scaled=pd.DataFrame(ms.fit_transform(x_encoded),columns=x_encoded.columns)
```

In [54]:

```
X_Scaled.head()
```

Out[54]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvemer
0	0.547619	0.716129	0.000000	0.25	0.000000	0.333333	0.914286	0.66666
1	0.738095	0.126165	0.250000	0.00	0.000484	0.666667	0.442857	0.33333
2	0.452381	0.910394	0.035714	0.25	0.001451	1.000000	0.885714	0.33333
3	0.357143	0.924014	0.071429	0.75	0.001935	1.000000	0.371429	0.66666
4	0.333333	0.646595	0.035714	0.25	0.003387	1.000000	0.700000	0.66666

5 rows × 50 columns

Splitting Dataset into Train and Test

In [55]:

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

In [56]:

```
X_train,X_test,y_train,y_test=train_test_split(X_Scaled,y_encoded,test_size=0.2,random_st
ate=0)
```

In [57]:

```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

Out[57]:

```
((721, 50), (181, 50), (721, 1), (181, 1))
```

• Model Building

In [58]:

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

In [59]:

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

```
C:\Users\kommi\anaconda3\lib\site-packages\sklearn\utils\validation.py:993: DataConversio
nWarning: A column-vector y was passed when a 1d array was expected. Please change the sh
ape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[59]:

```
LogisticRegression()
```

In [60]:

```
predict=model.predict(X_test)
```

In [61]:

```
predict
```

Out[61]:

```
array([0, 0, 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,
       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, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 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, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0,
       1, 0, 0, 0, 0], dtype=uint8)
```

In [62]:

```
y_test
```

Out[62]:

	Yes
399	0
1182	0
583	0
1290	1
1061	0
...	...
2	1
643	0
1292	0
1383	0
845	0

181 rows × 1 columns

In [63]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report,roc_auc
_score,roc_curve
```

In [64]:

```
accuracy_score(y_test,predict)
```

Out[64]:

```
0.8342541436464088
```

In [65]:

```
confusion_matrix(y_test,predict)
```

Out[65]:

```
array([[133,   7],
       [ 23,  18]], dtype=int64)
```

In [66]:

```
print(classification_report(y_test,predict))
```

```
              precision    recall  f1-score   support

           0       0.85      0.95      0.90       140
           1       0.72      0.44      0.55        41

    accuracy                           0.83       181
   macro avg       0.79      0.69      0.72       181
weighted avg       0.82      0.83      0.82       181
```

Decision Tree

In [67]:

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

In [68]:

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

Out[68]:

```
DecisionTreeClassifier()
```

In [69]:

```
Predict=dtc.predict(X_test)
```

In [70]:

```
Predict
```

Out[70]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
       1, 0, 0, 0, 0], dtype=uint8)
```

In [71]:

```
accuracy_score(y_test,Predict)
```

Out[71]:

```
0.7016574585635359
```

In [72]:

```
confusion_matrix(y_test,Predict)
```

Out[72]:

```
array([[111,  29],
       [ 25,  16]], dtype=int64)
```

In [73]:

```
print(classification_report(y_test,Predict))
```

```
              precision    recall  f1-score   support

           0       0.82      0.79      0.80       140
           1       0.36      0.39      0.37        41

    accuracy                           0.70       181
   macro avg       0.59      0.59      0.59       181
weighted avg       0.71      0.70      0.71       181
```

In [74]:

```
y_test=y_test['Yes']
```

In [75]:

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

Out[75]:

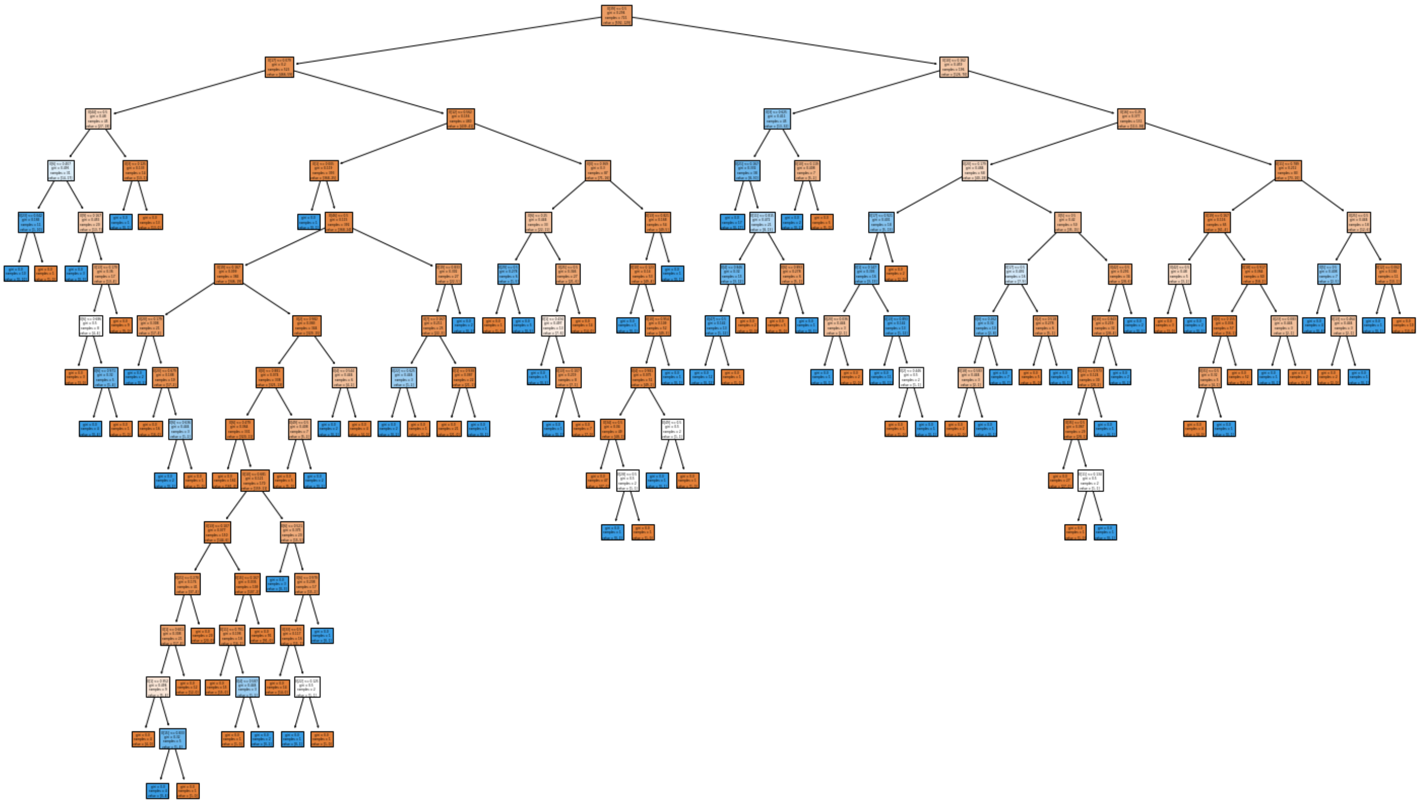
col_0	0	1
Yes		
0	111	29
1	25	16

In [76]:

```
from sklearn import tree
plt.figure(figsize=(25,15))
tree.plot_tree(dtc,filled=True)
```

Out[76]:

```
[Text(0.434375, 0.96875, 'X[39] <= 0.5\ngini = 0.294\nsamples = 721\nvalue = [592, 129]')
,
 Text(0.19638157894736843, 0.90625, 'X[17] <= 0.079\ngini = 0.2\nsamples = 525\nvalue = [
466, 59]'),
 Text(0.06842105263157895, 0.84375, 'X[44] <= 0.5\ngini = 0.48\nsamples = 45\nvalue = [27
, 18]'),
 Text(0.042105263157894736, 0.78125, 'X[6] <= 0.407\ngini = 0.495\nsamples = 31\nvalue =
[14, 17]'),
 Text(0.021052631578947368, 0.71875, 'X[23] <= 0.042\ngini = 0.165\nsamples = 11\nvalue =
[1, 10]'),
 Text(0.010526315789473684, 0.65625, 'gini = 0.0\nsamples = 10\nvalue = [0, 10]'),
 Text(0.031578947368421054, 0.65625, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.06315789473684211, 0.71875, 'X[9] <= 0.167\ngini = 0.455\nsamples = 20\nvalue = [
13, 7]'),
 Text(0.05263157894736842, 0.65625, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),
 Text(0.07368421052631578, 0.65625, 'X[13] <= 0.179\ngini = 0.36\nsamples = 17\nvalue = [
13, 4]'),
 Text(0.06315789473684211, 0.59375, 'X[6] <= 0.686\ngini = 0.5\nsamples = 8\nvalue = [4,
4]'),
 Text(0.05263157894736842, 0.53125, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),
 Text(0.07368421052631578, 0.53125, 'X[6] <= 0.971\ngini = 0.32\nsamples = 5\nvalue = [1,
4]'),
 Text(0.06315789473684211, 0.46875, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]'),
 Text(0.08421052631578947, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.08421052631578947, 0.59375, 'gini = 0.0\nsamples = 9\nvalue = [9, 0]'),
 Text(0.09473684210526316, 0.78125, 'X[3] <= 0.125\ngini = 0.133\nsamples = 14\nvalue = [
13, 1]'),
 Text(0.08421052631578947, 0.71875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.10526315789473684, 0.71875, 'gini = 0.0\nsamples = 13\nvalue = [13, 0]'),
 Text(0.3243421052631579, 0.84375, 'X[12] <= 0.562\ngini = 0.156\nsamples = 480\nvalue =
[439, 41]'),
 Text(0.22763157894736843, 0.78125, 'X[1] <= 0.005\ngini = 0.119\nsamples = 393\nvalue =
[368, 25]'),
 Text(0.21710526315789475, 0.71875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.2381578947368421, 0.71875, 'X[46] <= 0.5\ngini = 0.115\nsamples = 392\nvalue = [3
68, 24]'),
 Text(0.16052631578947368, 0.65625, 'X[19] <= 0.167\ngini = 0.099\nsamples = 365\nvalue =
[346, 19]'),
 Text(0.10526315789473684, 0.59375, 'X[20] <= 0.179\ngini = 0.308\nsamples = 21\nvalue =
[17, 4]'),
 Text(0.09473684210526316, 0.53125, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.11578947368421053, 0.53125, 'X[20] <= 0.679\ngini = 0.188\nsamples = 19\nvalue =
[17, 2]'),
 Text(0.10526315789473684, 0.46875, 'gini = 0.0\nsamples = 16\nvalue = [16, 0]'),
 Text(0.12631578947368421, 0.46875, 'X[6] <= 0.636\ngini = 0.444\nsamples = 3\nvalue = [1
, 2]'),
 Text(0.11578947368421053, 0.40625, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.1368421052631579, 0.40625, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.21578947368421053, 0.59375, 'X[2] <= 0.982\ngini = 0.083\nsamples = 344\nvalue =
[329, 15]'),
 Text(0.18947368421052632, 0.53125, 'X[0] <= 0.881\ngini = 0.074\nsamples = 338\nvalue =
[325, 13]'),
 Text(0.16842105263157894, 0.46875, 'X[6] <= 0.479\ngini = 0.064\nsamples = 331\nvalue =
[320, 11]'),
 Text(0.15789473684210525, 0.40625, 'gini = 0.0\nsamples = 161\nvalue = [161, 0]'),
 Text(0.17894736842105263, 0.40625, 'X[10] <= 0.681\ngini = 0.121\nsamples = 170\nvalue =
[159, 11]'),
 Text(0.15263157894736842, 0.34375, 'X[13] <= 0.107\ngini = 0.077\nsamples = 150\nvalue =
[144, 6]'),
 Text(0.13157894736842105, 0.28125, 'X[21] <= 0.278\ngini = 0.176\nsamples = 41\nvalue =
[37, 4]'),
 Text(0.12105263157894737, 0.21875, 'X[1] <= 0.601\ngini = 0.308\nsamples = 21\nvalue = [
17, 4]'),
 Text(0.11052631578947368, 0.15625, 'X[1] <= 0.352\ngini = 0.494\nsamples = 9\nvalue = [5
, 4]'),
 Text(0.1, 0.09375, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),
 Text(0.12105263157894737, 0.09375, 'X[15] <= 0.833\ngini = 0.32\nsamples = 5\nvalue = [1
, 4]'),
 Text(0.11052631578947368, 0.03125, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]'),
 Text(0.13157894736842105, 0.03125, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.13157894736842105, 0.15625, 'gini = 0.0\nsamples = 12\nvalue = [12, 0]'),
 Text(0.14210526315789473, 0.21875, 'gini = 0.0\nsamples = 20\nvalue = [20, 0]'),
 Text(0.1736842105263158, 0.28125, 'X[15] <= 0.167\ngini = 0.036\nsamples = 109\nvalue =
[107, 2]'),
 Text(0.1631578947368421, 0.21875, 'X[11] <= 0.791\ngini = 0.198\nsamples = 18\nvalue = [
16, 2]'),
 Text(0.15263157894736842, 0.15625, 'gini = 0.0\nsamples = 15\nvalue = [15, 0]'),
 Text(0.1736842105263158, 0.15625, 'X[4] <= 0.587\ngini = 0.444\nsamples = 3\nvalue = [1,
2]'),
 Text(0.1631578947368421, 0.09375, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.18421052631578946, 0.09375, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.18421052631578946, 0.21875, 'gini = 0.0\nsamples = 91\nvalue = [91, 0]'),
 Text(0.20526315789473684, 0.34375, 'X[6] <= 0.521\ngini = 0.375\nsamples = 20\nvalue = [
15, 5]'),
 Text(0.19473684210526315, 0.28125, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),
 Text(0.21578947368421053, 0.28125, 'X[6] <= 0.979\ngini = 0.208\nsamples = 17\nvalue = [
15, 2]'),
 Text(0.20526315789473684, 0.21875, 'X[33] <= 0.5\ngini = 0.117\nsamples = 16\nvalue = [1
5, 1]'),
 Text(0.19473684210526315, 0.15625, 'gini = 0.0\nsamples = 14\nvalue = [14, 0]'),
 Text(0.21578947368421053, 0.15625, 'X[22] <= 0.125\ngini = 0.5\nsamples = 2\nvalue = [1,
1]'),
 Text(0.20526315789473684, 0.09375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.22631578947368422, 0.09375, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.22631578947368422, 0.21875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.21052631578947367, 0.46875, 'X[49] <= 0.5\ngini = 0.408\nsamples = 7\nvalue = [5,
2]'),
 Text(0.2, 0.40625, 'gini = 0.0\nsamples = 5\nvalue = [5, 0]'),
 Text(0.22105263157894736, 0.40625, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.24210526315789474, 0.53125, 'X[4] <= 0.544\ngini = 0.444\nsamples = 6\nvalue = [4
, 2]'),
 Text(0.23157894736842105, 0.46875, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.25263157894736843, 0.46875, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),
 Text(0.3157894736842105, 0.65625, 'X[19] <= 0.833\ngini = 0.302\nsamples = 27\nvalue = [
22, 5]'),
 Text(0.30526315789473685, 0.59375, 'X[7] <= 0.167\ngini = 0.211\nsamples = 25\nvalue = [
22, 3]'),
 Text(0.28421052631578947, 0.53125, 'X[22] <= 0.625\ngini = 0.444\nsamples = 3\nvalue = [
1, 2]'),
 Text(0.2736842105263158, 0.46875, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.29473684210526313, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.3263157894736842, 0.53125, 'X[1] <= 0.938\ngini = 0.087\nsamples = 22\nvalue = [2
1, 1]'),
 Text(0.3157894736842105, 0.46875, 'gini = 0.0\nsamples = 21\nvalue = [21, 0]'),
 Text(0.3368421052631579, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.3263157894736842, 0.59375, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.42105263157894735, 0.78125, 'X[0] <= 0.369\ngini = 0.3\nsamples = 87\nvalue = [71
, 16]'),
 Text(0.37894736842105264, 0.71875, 'X[6] <= 0.25\ngini = 0.444\nsamples = 33\nvalue = [2
2, 11]'),
 Text(0.35789473684210527, 0.65625, 'X[29] <= 0.5\ngini = 0.278\nsamples = 6\nvalue = [1,
5]'),
 Text(0.3473684210526316, 0.59375, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.3684210526315789, 0.59375, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),
 Text(0.4, 0.65625, 'X[25] <= 0.5\ngini = 0.346\nsamples = 27\nvalue = [21, 6]'),
 Text(0.3894736842105263, 0.59375, 'X[1] <= 0.434\ngini = 0.497\nsamples = 13\nvalue = [7
, 6]'),
 Text(0.37894736842105264, 0.53125, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),
 Text(0.4, 0.53125, 'X[13] <= 0.107\ngini = 0.219\nsamples = 8\nvalue = [7, 1]'),
 Text(0.3894736842105263, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.4105263157894737, 0.46875, 'gini = 0.0\nsamples = 7\nvalue = [7, 0]'),
 Text(0.4105263157894737, 0.59375, 'gini = 0.0\nsamples = 14\nvalue = [14, 0]'),
 Text(0.4631578947368421, 0.71875, 'X[13] <= 0.821\ngini = 0.168\nsamples = 54\nvalue = [
49, 5]'),
 Text(0.45263157894736844, 0.65625, 'X[10] <= 0.123\ngini = 0.14\nsamples = 53\nvalue = [
49, 4]'),
 Text(0.4421052631578947, 0.59375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.4631578947368421, 0.59375, 'X[10] <= 0.954\ngini = 0.109\nsamples = 52\nvalue = [
49, 3]'),
 Text(0.45263157894736844, 0.53125, 'X[4] <= 0.981\ngini = 0.075\nsamples = 51\nvalue = [
49, 2]'),
 Text(0.43157894736842106, 0.46875, 'X[34] <= 0.5\ngini = 0.04\nsamples = 49\nvalue = [48
, 1]'),
 Text(0.42105263157894735, 0.40625, 'gini = 0.0\nsamples = 47\nvalue = [47, 0]'),
 Text(0.4421052631578947, 0.40625, 'X[28] <= 0.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]
'),
 Text(0.43157894736842106, 0.34375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.45263157894736844, 0.34375, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.47368421052631576, 0.46875, 'X[49] <= 0.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1
]'),
 Text(0.4631578947368421, 0.40625, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.4842105263157895, 0.40625, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.47368421052631576, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.47368421052631576, 0.65625, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.6723684210526316, 0.90625, 'X[10] <= 0.162\ngini = 0.459\nsamples = 196\nvalue =
[126, 70]'),
 Text(0.5473684210526316, 0.84375, 'X[3] <= 0.625\ngini = 0.411\nsamples = 45\nvalue = [1
3, 32]'),
 Text(0.5263157894736842, 0.78125, 'X[21] <= 0.167\ngini = 0.332\nsamples = 38\nvalue = [
8, 30]'),
 Text(0.5157894736842106, 0.71875, 'gini = 0.0\nsamples = 17\nvalue = [0, 17]'),
 Text(0.5368421052631579, 0.71875, 'X[11] <= 0.815\ngini = 0.472\nsamples = 21\nvalue = [
8, 13]'),
 Text(0.5157894736842106, 0.65625, 'X[4] <= 0.846\ngini = 0.32\nsamples = 15\nvalue = [3,
12]'),
 Text(0.5052631578947369, 0.59375, 'X[47] <= 0.5\ngini = 0.142\nsamples = 13\nvalue = [1,
12]'),
 Text(0.49473684210526314, 0.53125, 'gini = 0.0\nsamples = 12\nvalue = [0, 12]'),
 Text(0.5157894736842106, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.5263157894736842, 0.59375, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.5578947368421052, 0.65625, 'X[6] <= 0.893\ngini = 0.278\nsamples = 6\nvalue = [5,
1]'),
 Text(0.5473684210526316, 0.59375, 'gini = 0.0\nsamples = 5\nvalue = [5, 0]'),
 Text(0.5684210526315789, 0.59375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.5684210526315789, 0.78125, 'X[10] <= 0.119\ngini = 0.408\nsamples = 7\nvalue = [5
, 2]'),
 Text(0.5578947368421052, 0.71875, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.5789473684210527, 0.71875, 'gini = 0.0\nsamples = 5\nvalue = [5, 0]'),
 Text(0.7973684210526316, 0.84375, 'X[16] <= 0.25\ngini = 0.377\nsamples = 151\nvalue = [
113, 38]'),
 Text(0.6868421052631579, 0.78125, 'X[20] <= 0.179\ngini = 0.484\nsamples = 68\nvalue = [
40, 28]'),
 Text(0.6210526315789474, 0.71875, 'X[17] <= 0.921\ngini = 0.401\nsamples = 18\nvalue = [
5, 13]'),
 Text(0.6105263157894737, 0.65625, 'X[1] <= 0.147\ngini = 0.305\nsamples = 16\nvalue = [3
, 13]'),
 Text(0.5894736842105263, 0.59375, 'X[20] <= 0.036\ngini = 0.444\nsamples = 3\nvalue = [2
, 1]'),
 Text(0.5789473684210527, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.6, 0.53125, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.631578947368421, 0.59375, 'X[13] <= 0.893\ngini = 0.142\nsamples = 13\nvalue = [1
, 12]'),
 Text(0.6210526315789474, 0.53125, 'gini = 0.0\nsamples = 11\nvalue = [0, 11]'),
 Text(0.6421052631578947, 0.53125, 'X[2] <= 0.446\ngini = 0.5\nsamples = 2\nvalue = [1, 1
]'),
 Text(0.631578947368421, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.6526315789473685, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.631578947368421, 0.65625, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.7526315789473684, 0.71875, 'X[5] <= 0.5\ngini = 0.42\nsamples = 50\nvalue = [35,
15]'),
 Text(0.7157894736842105, 0.65625, 'X[17] <= 0.5\ngini = 0.492\nsamples = 16\nvalue = [7,
9]'),
 Text(0.6947368421052632, 0.59375, 'X[0] <= 0.202\ngini = 0.32\nsamples = 10\nvalue = [2,
8]'),
 Text(0.6842105263157895, 0.53125, 'X[18] <= 0.583\ngini = 0.444\nsamples = 3\nvalue = [2
, 1]'),
 Text(0.6736842105263158, 0.46875, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.6947368421052632, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.7052631578947368, 0.53125, 'gini = 0.0\nsamples = 7\nvalue = [0, 7]'),
 Text(0.7368421052631579, 0.59375, 'X[2] <= 0.518\ngini = 0.278\nsamples = 6\nvalue = [5,
1]'),
 Text(0.7263157894736842, 0.53125, 'gini = 0.0\nsamples = 5\nvalue = [5, 0]'),
 Text(0.7473684210526316, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.7894736842105263, 0.65625, 'X[42] <= 0.5\ngini = 0.291\nsamples = 34\nvalue = [28
, 6]'),
 Text(0.7789473684210526, 0.59375, 'X[10] <= 0.843\ngini = 0.219\nsamples = 32\nvalue = [
28, 4]'),
 Text(0.7684210526315789, 0.53125, 'X[11] <= 0.973\ngini = 0.124\nsamples = 30\nvalue = [
28, 2]'),
 Text(0.7578947368421053, 0.46875, 'X[35] <= 0.5\ngini = 0.067\nsamples = 29\nvalue = [28
, 1]'),
 Text(0.7473684210526316, 0.40625, 'gini = 0.0\nsamples = 27\nvalue = [27, 0]'),
 Text(0.7684210526315789, 0.40625, 'X[11] <= 0.134\ngini = 0.5\nsamples = 2\nvalue = [1,
1]'),
 Text(0.7578947368421053, 0.34375, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),
 Text(0.7789473684210526, 0.34375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.7789473684210526, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.7894736842105263, 0.53125, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.8, 0.59375, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.9078947368421053, 0.78125, 'X[11] <= 0.789\ngini = 0.212\nsamples = 83\nvalue = [
73, 10]'),
 Text(0.8578947368421053, 0.71875, 'X[19] <= 0.167\ngini = 0.116\nsamples = 65\nvalue = [
61, 4]'),
 Text(0.8315789473684211, 0.65625, 'X[42] <= 0.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2
]'),
 Text(0.8210526315789474, 0.59375, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),
 Text(0.8421052631578947, 0.59375, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),
 Text(0.8842105263157894, 0.65625, 'X[18] <= 0.917\ngini = 0.064\nsamples = 60\nvalue = [
58, 2]'),
 Text(0.8631578947368421, 0.59375, 'X[0] <= 0.155\ngini = 0.034\nsamples = 57\nvalue = [5
6, 1]'),
 Text(0.8526315789473684, 0.53125, 'X[31] <= 0.5\ngini = 0.32\nsamples = 5\nvalue = [4, 1
]'),
 Text(0.8421052631578947, 0.46875, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),
 Text(0.8631578947368421, 0.46875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.8736842105263158, 0.53125, 'gini = 0.0\nsamples = 52\nvalue = [52, 0]'),
 Text(0.9052631578947369, 0.59375, 'X[23] <= 0.083\ngini = 0.444\nsamples = 3\nvalue = [2
, 1]'),
 Text(0.8947368421052632, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.9157894736842105, 0.53125, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.9578947368421052, 0.71875, 'X[25] <= 0.5\ngini = 0.444\nsamples = 18\nvalue = [12
, 6]'),
 Text(0.9368421052631579, 0.65625, 'X[5] <= 0.5\ngini = 0.408\nsamples = 7\nvalue = [2, 5
]'),
 Text(0.9263157894736842, 0.59375, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]'),
 Text(0.9473684210526315, 0.59375, 'X[13] <= 0.464\ngini = 0.444\nsamples = 3\nvalue = [2
, 1]'),
 Text(0.9368421052631579, 0.53125, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),
 Text(0.9578947368421052, 0.53125, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.9789473684210527, 0.65625, 'X[12] <= 0.062\ngini = 0.165\nsamples = 11\nvalue = [
10, 1]'),
 Text(0.968421052631579, 0.59375, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),
 Text(0.9894736842105263, 0.59375, 'gini = 0.0\nsamples = 10\nvalue = [10, 0]')]
```



In [77]:

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

}
```

In [78]:

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

In [79]:

```
grid_search.fit(X_train,y_train)
```

Out[79]:

Out[79]:

```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [1, 2, 3, 4, 5],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'splitter': ['best']},
             scoring='accuracy')
```

In [80]:

```
grid_search.best_params_
```

Out[80]:

```
{'criterion': 'entropy',
 'max_depth': 3,
 'max_features': 'log2',
 'splitter': 'best'}
```

In [81]:

```
dtc_cv=DecisionTreeClassifier(criterion= 'entropy',
 max_depth=3,
 max_features='sqrt',
 splitter='best')
dtc_cv.fit(X_train,y_train)
```

Out[81]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=3, max_features='sqrt')
```

In [82]:

```
predict=dtc_cv.predict(X_test)
```

In [83]:

```
print(classification_report(y_test,predict))
```

```
              precision    recall  f1-score   support

           0       0.78      0.99      0.87       140
           1       0.50      0.05      0.09        41

    accuracy                           0.77       181
   macro avg       0.64      0.52      0.48       181
weighted avg       0.72      0.77      0.69       181
```

In [85]:

```
accuracy_score(y_test,predict)
```

Out[85]:

```
0.7734806629834254
```

Random Forest

In [86]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [87]:

```
random_forest = RandomForestClassifier(n_estimators=100, random_state=42)
```

In [88]:

```
random_forest.fit(X_train, y_train)
```

```
C:\Users\kommi\AppData\Local\Temp\ipykernel_3576\2566748861.py:1: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y t
o (n_samples,), for example using ravel().
  random_forest.fit(X_train, y_train)
```

Out[88]:

```
RandomForestClassifier(random_state=42)
```

In [89]:

```
y_pred = random_forest.predict(X_test)
```

In [90]:

```
y_pred
```

Out[90]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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, 1, 0, 0, 0, 1, 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, 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, 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, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0], dtype=uint8)
```

In [91]:

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

```
0.8066298342541437
```

In [92]:

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

```
              precision    recall  f1-score   support

           0       0.81      0.99      0.89       140
           1       0.80      0.20      0.31        41

    accuracy                           0.81       181
   macro avg       0.80      0.59      0.60       181
weighted avg       0.81      0.81      0.76       181
```