employee-attrition-assignment

September 28, 2023

0.1 Employee Attrition Assignment

Name: CH.Ganesh Sri Naga Venkata Ajay

Reg.No: 21BDS0269

Link : https://www.kaggle.com/datasets/patelprashant/employee-attrition

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

[2]:
```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

[3]:
```
df = pd.read_csv("/content/WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

[4]:
```
df.head(3)
```

[4]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development

   DistanceFromHome  Education EducationField  EmployeeCount  EmployeeNumber  \
0                 1          2  Life Sciences              1               1
1                 8          1  Life Sciences              1               2
2                 2          2          Other              1               4

   …  RelationshipSatisfaction StandardHours  StockOptionLevel  \
0  …                         1            80                 0
1  …                         4            80                 1
2  …                         2            80                 0

   TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                  8                      0               1               6
1                 10                      3               3              10
2                  7                      3               3               0
```

```
  YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                  4                        0                     5
1                  7                        1                     7
2                  0                        0                     0

[3 rows x 35 columns]
```

0.2 About DataSet:

Output Variable: Attrition

Education: The highest level of education. 1: Below College, 2:College, 3: Bachelor, 4: Master, 5: Doctor.

EnvironmentSatisfaction: A rating of work environment. 1: Low, 2: Medium, 3: High, 4: Very High.

JobInvolvement: Level of job involvement. 1: Low, 2: Medium, 3: High, 4: Very High.

PerformanceRating: Performance rating. 1: Low, 2: Good, 3: Excellent, 4: Outstanding.

RelationshipSatisfaction: Relationship satisfaction among at workplace and family members. 1: Low, 2: Medium, 3: High, 4: Very High.

WorkLifeBalance: Work life balance rating. 1: Bad, 2: Good, 3: Better, 4: Best.

[5]:
```
df.shape
```

[5]:
```
(1470, 35)
```

[6]:
```
df.Attrition.value_counts()
```

[6]:
```
No     1233
Yes     237
Name: Attrition, dtype: int64
```

[7]:
```
df_info = pd.DataFrame(df.dtypes, columns=['dtypes'])
```

[8]:
```
df_info[df_info["dtypes"] == "object"].T.columns.tolist()
```

[8]:
```
['Attrition',
 'BusinessTravel',
 'Department',
 'EducationField',
 'Gender',
 'JobRole',
 'MaritalStatus',
 'Over18',
 'OverTime']
```

[9]:
```
df_info[df_info["dtypes"] == "int64"].T.columns.tolist()
```

[9]:
```
['Age',
 'DailyRate',
 'DistanceFromHome',
 'Education',
 'EmployeeCount',
 'EmployeeNumber',
 'EnvironmentSatisfaction',
 'HourlyRate',
 'JobInvolvement',
 'JobLevel',
 'JobSatisfaction',
 'MonthlyIncome',
 'MonthlyRate',
 'NumCompaniesWorked',
 'PercentSalaryHike',
 'PerformanceRating',
 'RelationshipSatisfaction',
 'StandardHours',
 'StockOptionLevel',
 'TotalWorkingYears',
 'TrainingTimesLastYear',
 'WorkLifeBalance',
 'YearsAtCompany',
 'YearsInCurrentRole',
 'YearsSinceLastPromotion',
 'YearsWithCurrManager']
```

[10]:
```
df.describe().T
```

[10]:
```
                           count          mean          std     min      25%  \
Age                       1470.0     36.923810     9.135373    18.0    30.00
DailyRate                 1470.0    802.485714   403.509100   102.0   465.00
DistanceFromHome          1470.0      9.192517     8.106864     1.0     2.00
Education                 1470.0      2.912925     1.024165     1.0     2.00
EmployeeCount             1470.0      1.000000     0.000000     1.0     1.00
EmployeeNumber            1470.0   1024.865306   602.024335     1.0   491.25
EnvironmentSatisfaction   1470.0      2.721769     1.093082     1.0     2.00
HourlyRate                1470.0     65.891156    20.329428    30.0    48.00
JobInvolvement            1470.0      2.729932     0.711561     1.0     2.00
JobLevel                  1470.0      2.063946     1.106940     1.0     1.00
JobSatisfaction           1470.0      2.728571     1.102846     1.0     2.00
MonthlyIncome             1470.0   6502.931293  4707.956783  1009.0  2911.00
MonthlyRate               1470.0  14313.103401  7117.786044  2094.0  8047.00
NumCompaniesWorked        1470.0      2.693197     2.498009     0.0     1.00
PercentSalaryHike         1470.0     15.209524     3.659938    11.0    12.00
```

PerformanceRating	1470.0	3.153741	0.360824	3.0	3.00
RelationshipSatisfaction	1470.0	2.712245	1.081209	1.0	2.00
StandardHours	1470.0	80.000000	0.000000	80.0	80.00
StockOptionLevel	1470.0	0.793878	0.852077	0.0	0.00
TotalWorkingYears	1470.0	11.279592	7.780782	0.0	6.00
TrainingTimesLastYear	1470.0	2.799320	1.289271	0.0	2.00
WorkLifeBalance	1470.0	2.761224	0.706476	1.0	2.00
YearsAtCompany	1470.0	7.008163	6.126525	0.0	3.00
YearsInCurrentRole	1470.0	4.229252	3.623137	0.0	2.00
YearsSinceLastPromotion	1470.0	2.187755	3.222430	0.0	0.00
YearsWithCurrManager	1470.0	4.123129	3.568136	0.0	2.00

	50%	75%	max
Age	36.0	43.00	60.0
DailyRate	802.0	1157.00	1499.0
DistanceFromHome	7.0	14.00	29.0
Education	3.0	4.00	5.0
EmployeeCount	1.0	1.00	1.0
EmployeeNumber	1020.5	1555.75	2068.0
EnvironmentSatisfaction	3.0	4.00	4.0
HourlyRate	66.0	83.75	100.0
JobInvolvement	3.0	3.00	4.0
JobLevel	2.0	3.00	5.0
JobSatisfaction	3.0	4.00	4.0
MonthlyIncome	4919.0	8379.00	19999.0
MonthlyRate	14235.5	20461.50	26999.0
NumCompaniesWorked	2.0	4.00	9.0
PercentSalaryHike	14.0	18.00	25.0
PerformanceRating	3.0	3.00	4.0
RelationshipSatisfaction	3.0	4.00	4.0
StandardHours	80.0	80.00	80.0
StockOptionLevel	1.0	1.00	3.0
TotalWorkingYears	10.0	15.00	40.0
TrainingTimesLastYear	3.0	3.00	6.0
WorkLifeBalance	3.0	3.00	4.0
YearsAtCompany	5.0	9.00	40.0
YearsInCurrentRole	3.0	7.00	18.0
YearsSinceLastPromotion	1.0	3.00	15.0
YearsWithCurrManager	3.0	7.00	17.0

0.2.1 Check For Null Values

[11]:
```
df.isnull().sum()
```

[11]:
```
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 in the dataset

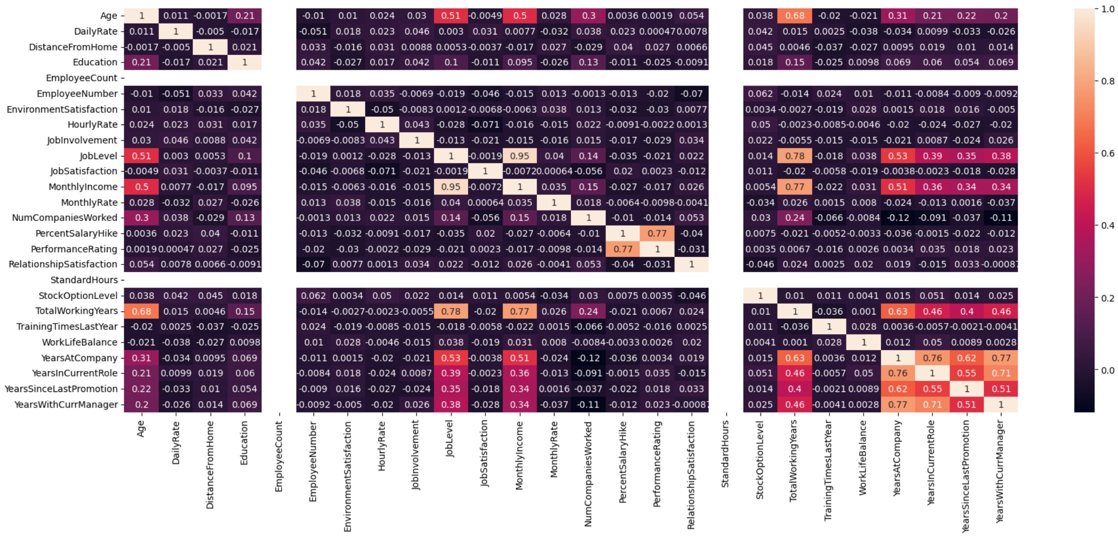
0.2.2 Data Visualization

[12]:
```python
print(df['EmployeeCount'].nunique())
print(df['StandardHours'].nunique())
```

```
1
1
```

[13]:
```python
plt.figure(figsize=(22, 8))
sns.heatmap(df.corr(numeric_only=True),annot=True)
```

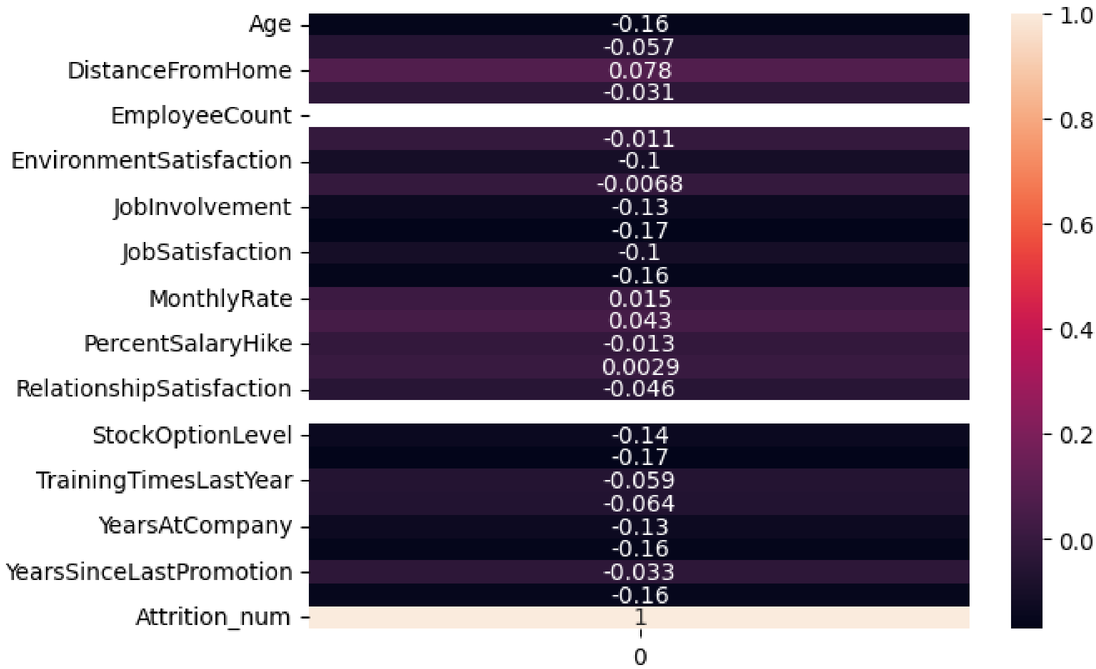
[13]: <Axes: >



- Here we are getting white space for both EmployeeCount and StandardHours because the values in all rows for them they have same values , if any column having all same values then Standard deviation will be Zero

[14]:
```
df['Attrition_num'] = df['Attrition'].map({'Yes': 1, 'No': 0})
correlation = df.corrwith(df['Attrition_num'])
sns.heatmap(correlation.to_frame(), annot=True)
```

[14]: <Axes: >



- Converted Yes:1 and No:1 in Attrition and added a new Column Attrition_num
- But the above correlation may be correct or not because we had converted categorical to numerical so we cannnot perfectly say that correlation was correct.

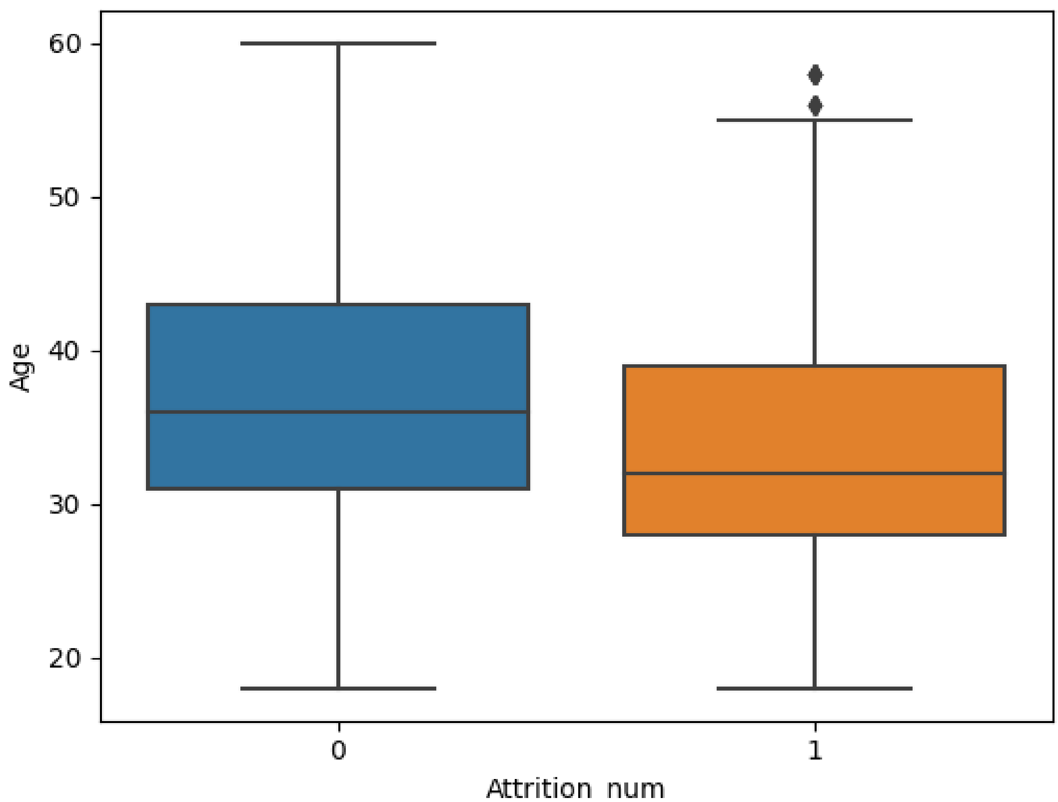
[15]:
```
corr_sort = correlation.abs().sort_values(ascending=False)
corr_sort
```

[15]:
```
Attrition_num               1.000000
TotalWorkingYears           0.171063
JobLevel                    0.169105
YearsInCurrentRole          0.160545
MonthlyIncome               0.159840
Age                         0.159205
YearsWithCurrManager        0.156199
StockOptionLevel            0.137145
YearsAtCompany              0.134392
JobInvolvement              0.130016
JobSatisfaction             0.103481
EnvironmentSatisfaction     0.103369
DistanceFromHome            0.077924
WorkLifeBalance             0.063939
TrainingTimesLastYear       0.059478
DailyRate                   0.056652
RelationshipSatisfaction    0.045872
NumCompaniesWorked          0.043494
```

```
YearsSinceLastPromotion     0.033019
Education                   0.031373
MonthlyRate                 0.015170
PercentSalaryHike           0.013478
EmployeeNumber              0.010577
HourlyRate                  0.006846
PerformanceRating           0.002889
EmployeeCount                    NaN
StandardHours                    NaN
dtype: float64
```

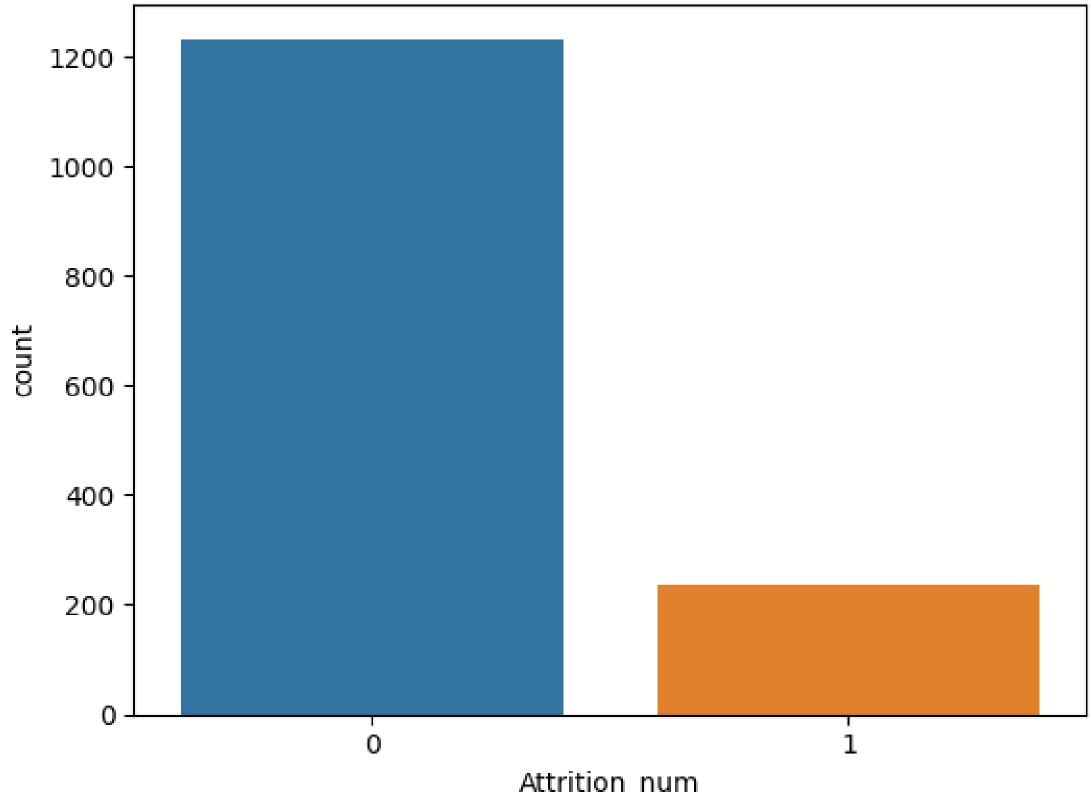
[16]:
```
sns.boxplot(x="Attrition_num",y="Age",data=df)
```

[16]: <Axes: xlabel='Attrition_num', ylabel='Age'>



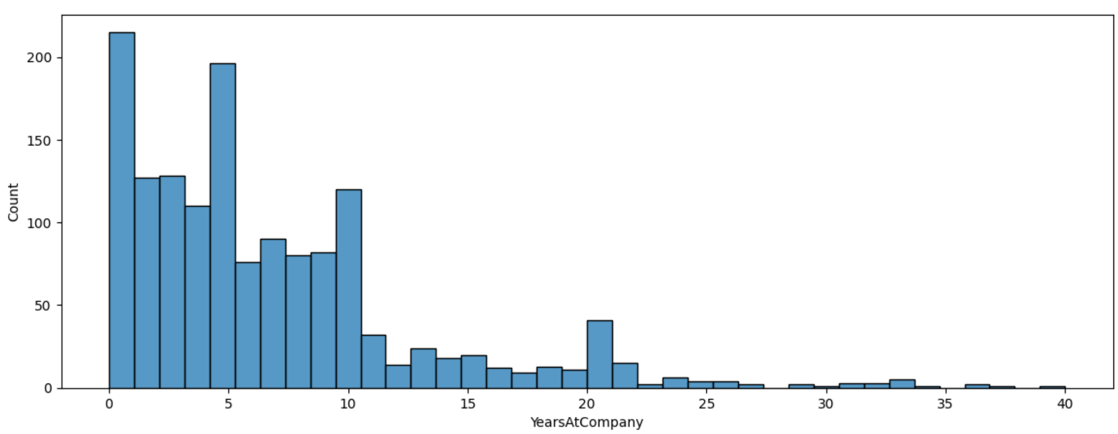
[17]:
```
sns.countplot(x=df["Attrition_num"])
```

[17]: <Axes: xlabel='Attrition_num', ylabel='count'>



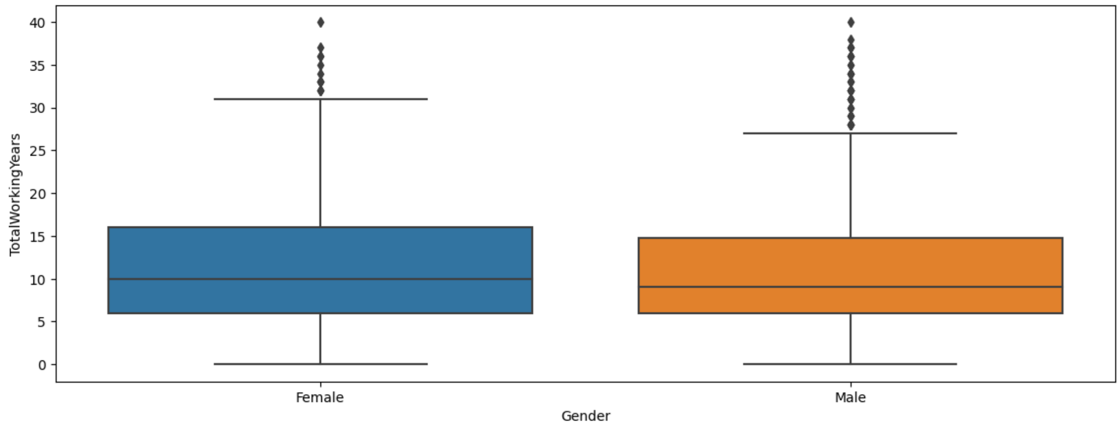

```
[18]: plt.figure(figsize=(14,5))
      sns.histplot(df["YearsAtCompany"])
```

```
[18]: <Axes: xlabel='YearsAtCompany', ylabel='Count'>
```



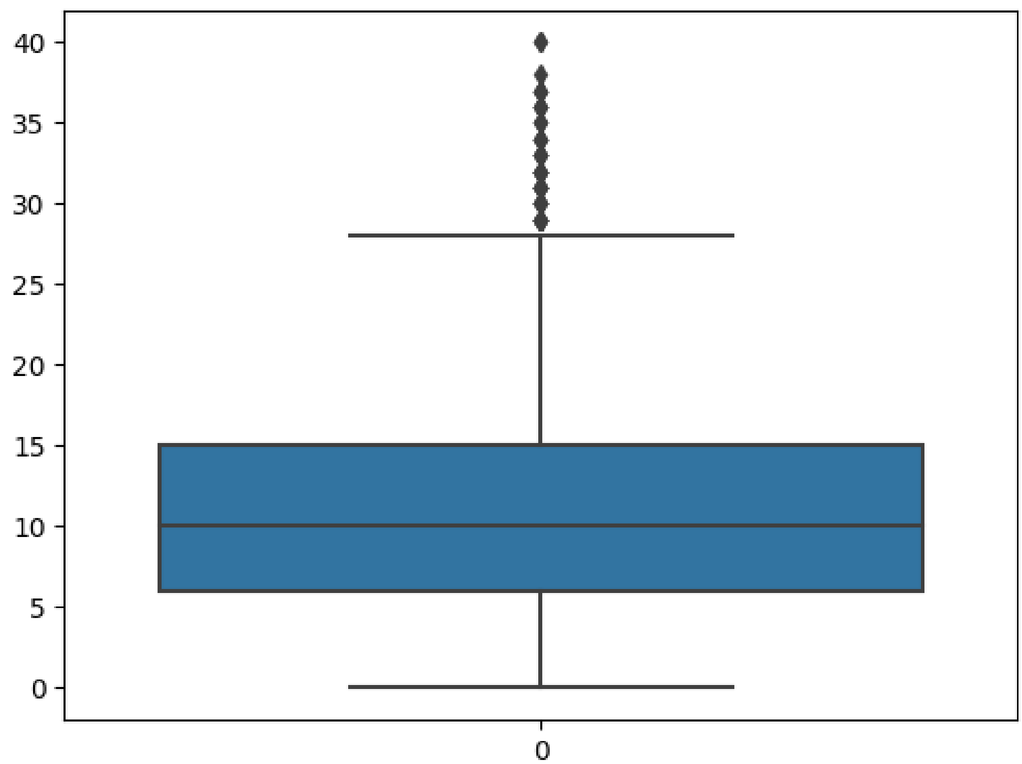
[19]:
```
plt.figure(figsize=(14,5))
sns.boxplot(y = df["TotalWorkingYears"],x=df["Gender"])
```

[19]: <Axes: xlabel='Gender', ylabel='TotalWorkingYears'>



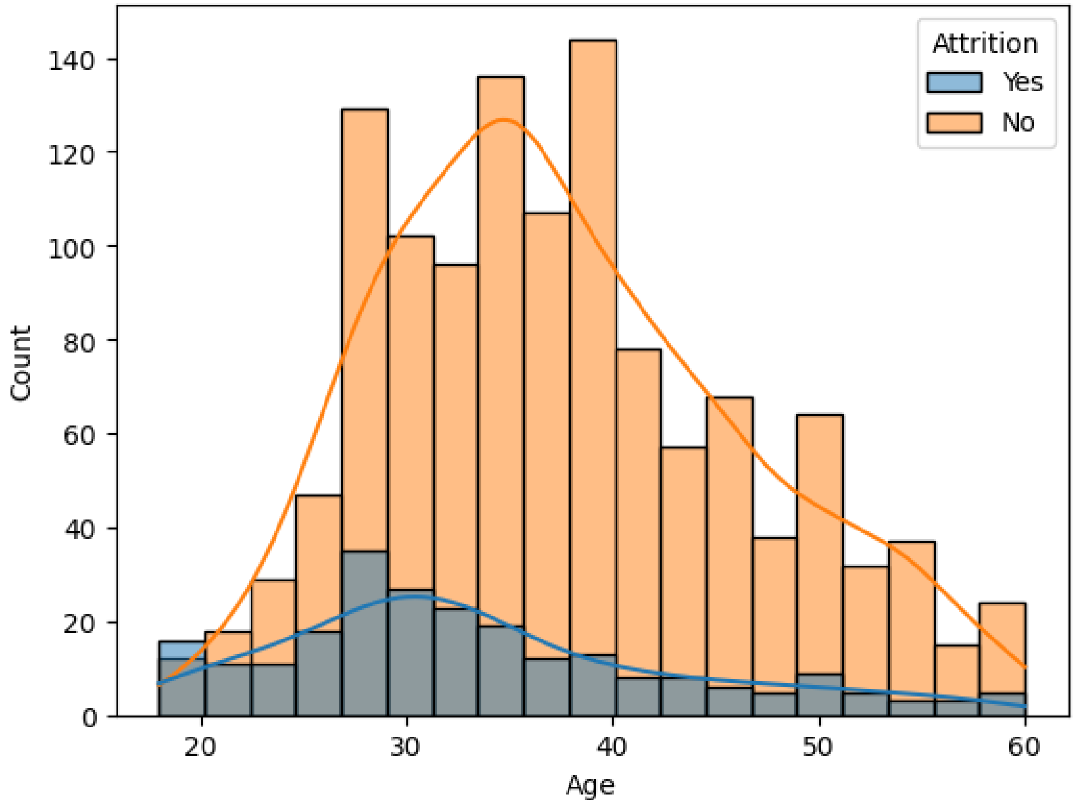
[20]:
```
sns.boxplot(df["TotalWorkingYears"])
```

[20]: <Axes: >



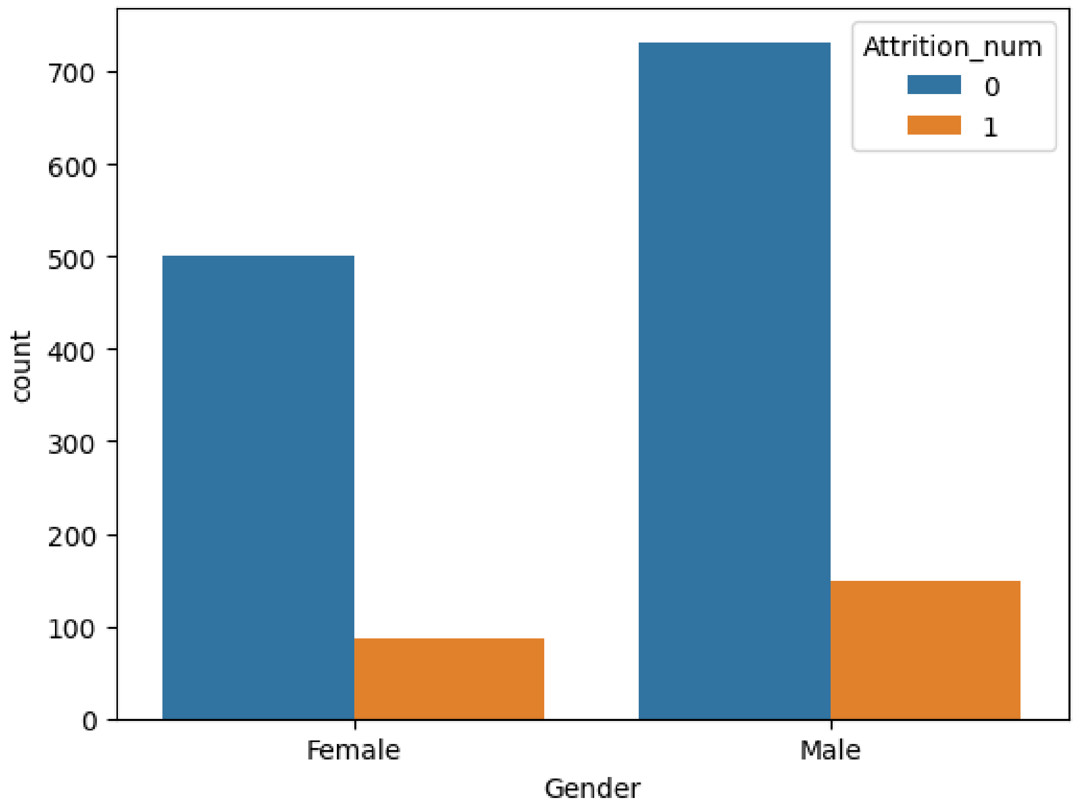
[21]:
```
sns.histplot(data=df, x='Age', hue='Attrition', kde=True)
```

[21]: <Axes: xlabel='Age', ylabel='Count'>



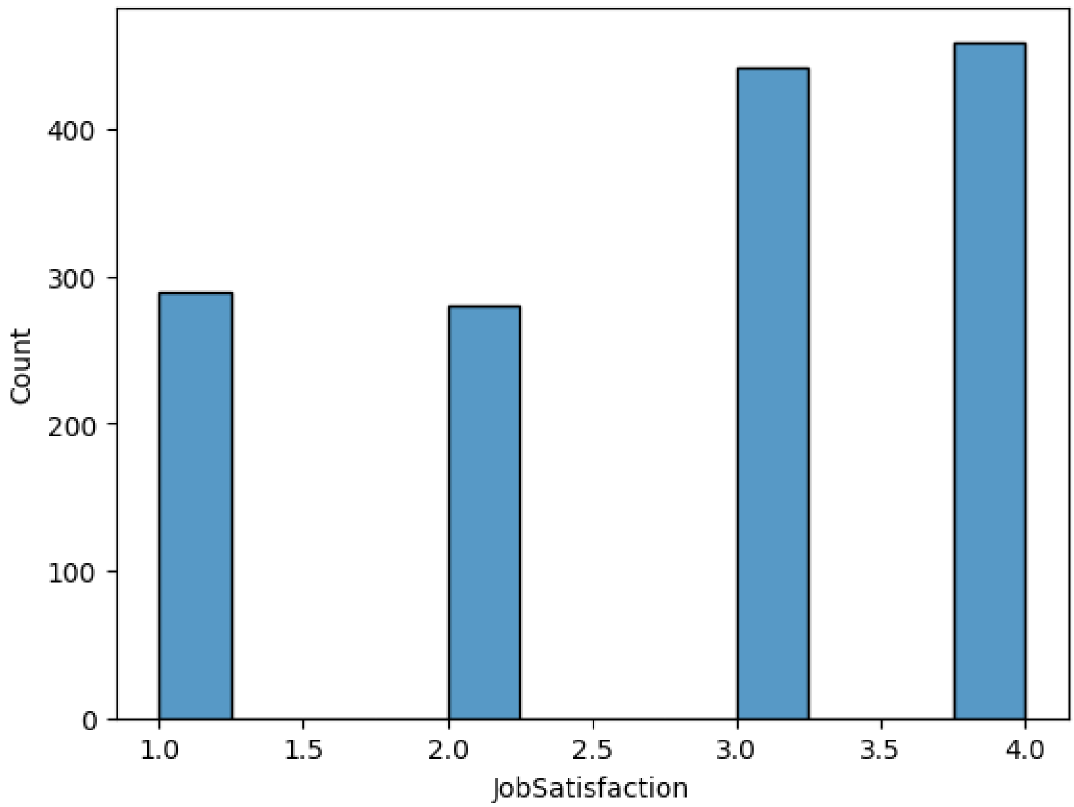
[22]:
```
sns.countplot(data=df, x='Gender', hue='Attrition_num')
```

[22]: <Axes: xlabel='Gender', ylabel='count'>



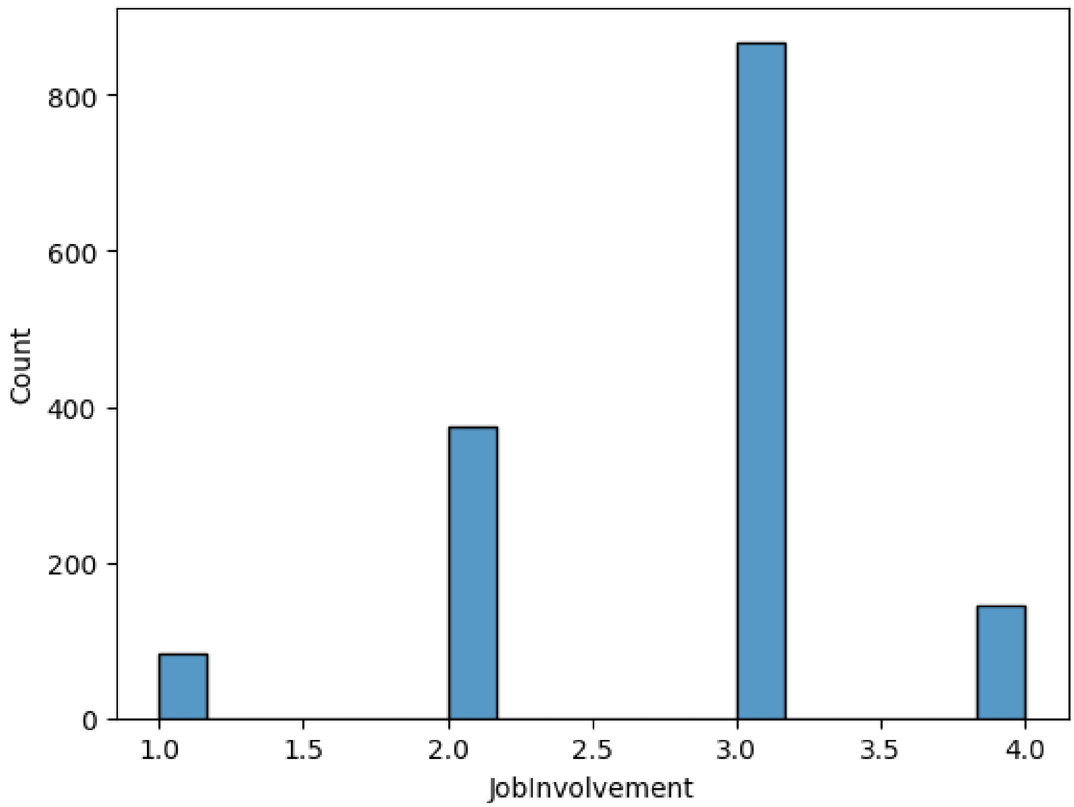
[23]:
```
sns.histplot(df["JobSatisfaction"])
```

[23]: <Axes: xlabel='JobSatisfaction', ylabel='Count'>



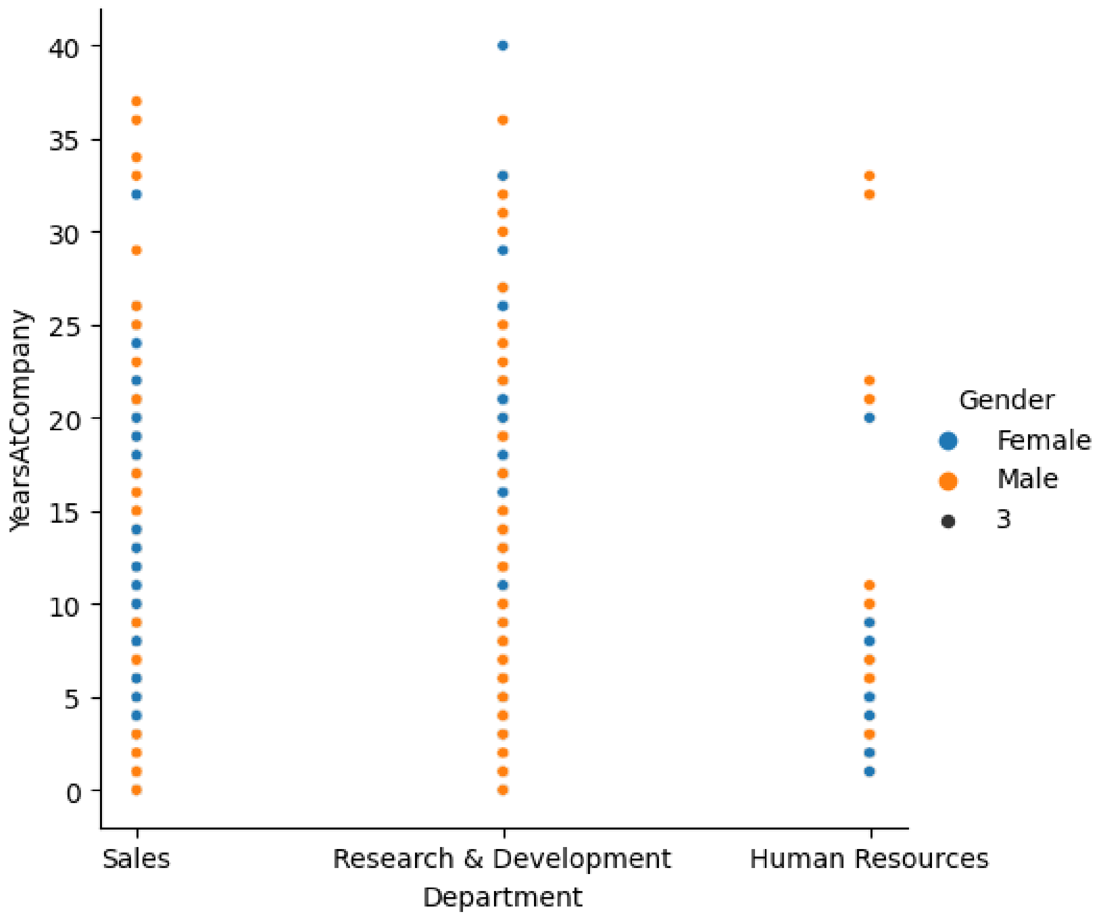
[24]:
```
sns.histplot(df["JobInvolvement"])
```

[24]: <Axes: xlabel='JobInvolvement', ylabel='Count'>



[25]:
```
sns.relplot(x="Department", y="YearsAtCompany",hue="Gender", data=df, size=3)
```

[25]:
```
<seaborn.axisgrid.FacetGrid at 0x7ad3df007a60>
```



sns.pairplot(df, height=8) plt.show()

0.2.3 Drop Unecessary Columns :

- EmployeeNumber, as it was diff for all so dropping is best. This was unique id to all, model will never depend on this
- EmployeeCount, StandardHours -> same for all so no need
- Over18 only have Y so removing is best

[26]:
```
df['Over18'].unique()
```

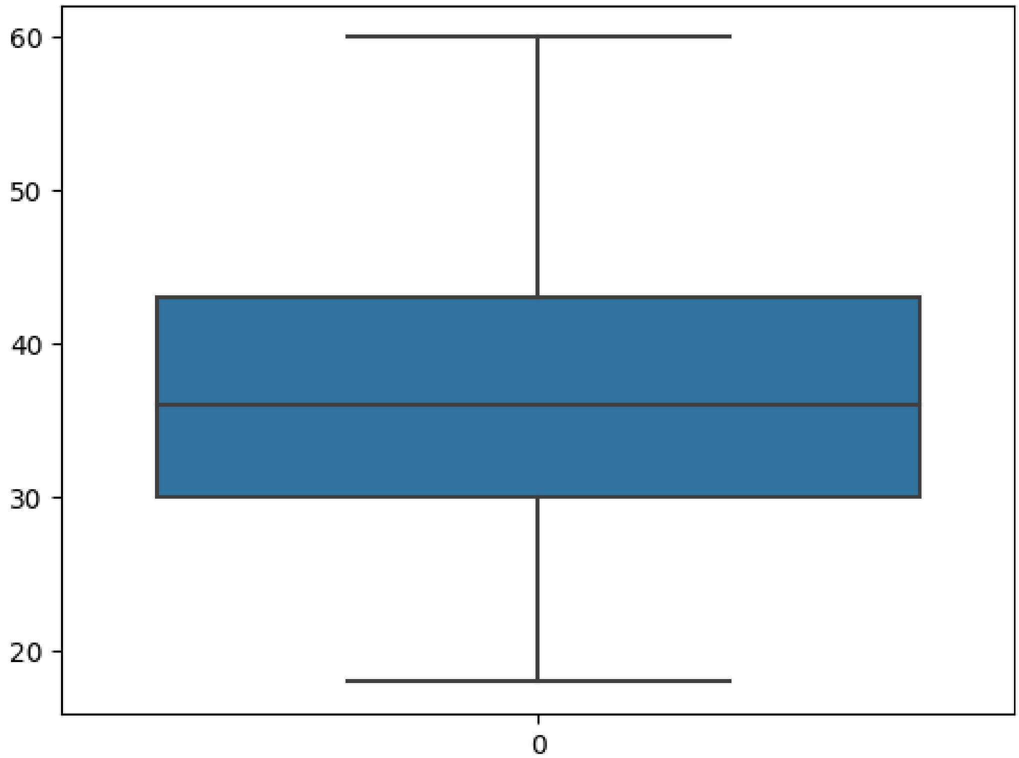
[26]:
```
array(['Y'], dtype=object)
```

[27]:
```
df.
 ↪drop(columns=["EmployeeNumber","StandardHours","EmployeeCount","Over18"],axis=1,inplace=Tru
```

0.2.4 Outliers

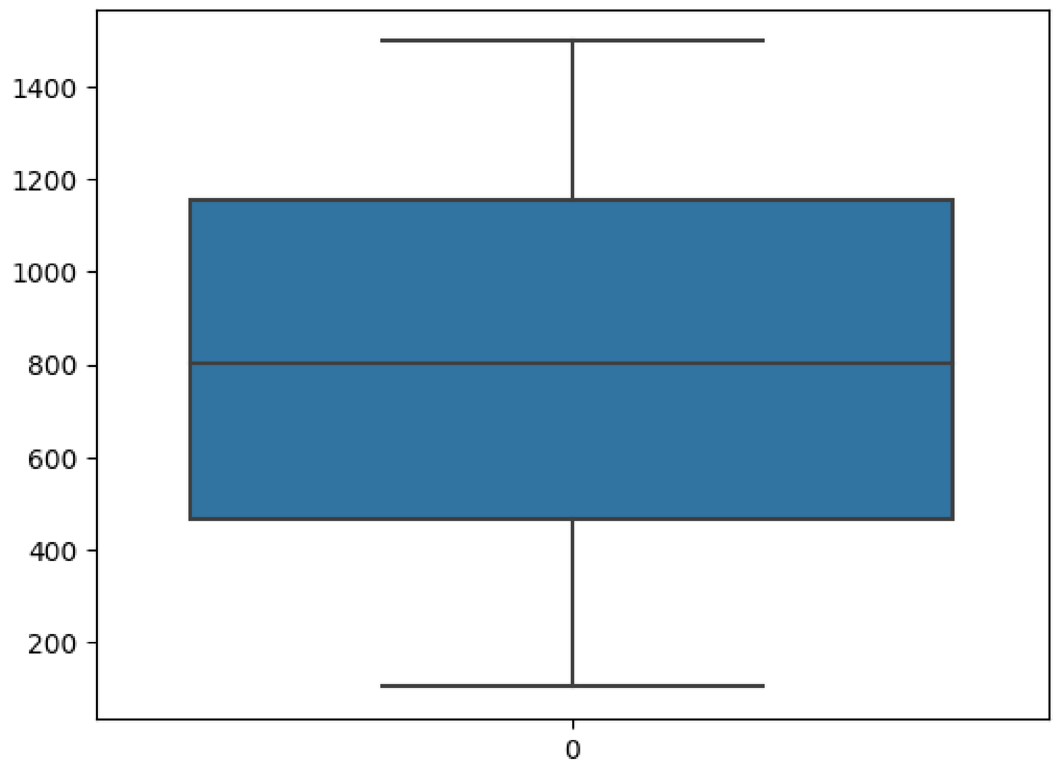
[28]:
```
sns.boxplot(df.Age)
```

[28]: <Axes: >



[29]:
```
sns.boxplot(df.DailyRate)
```

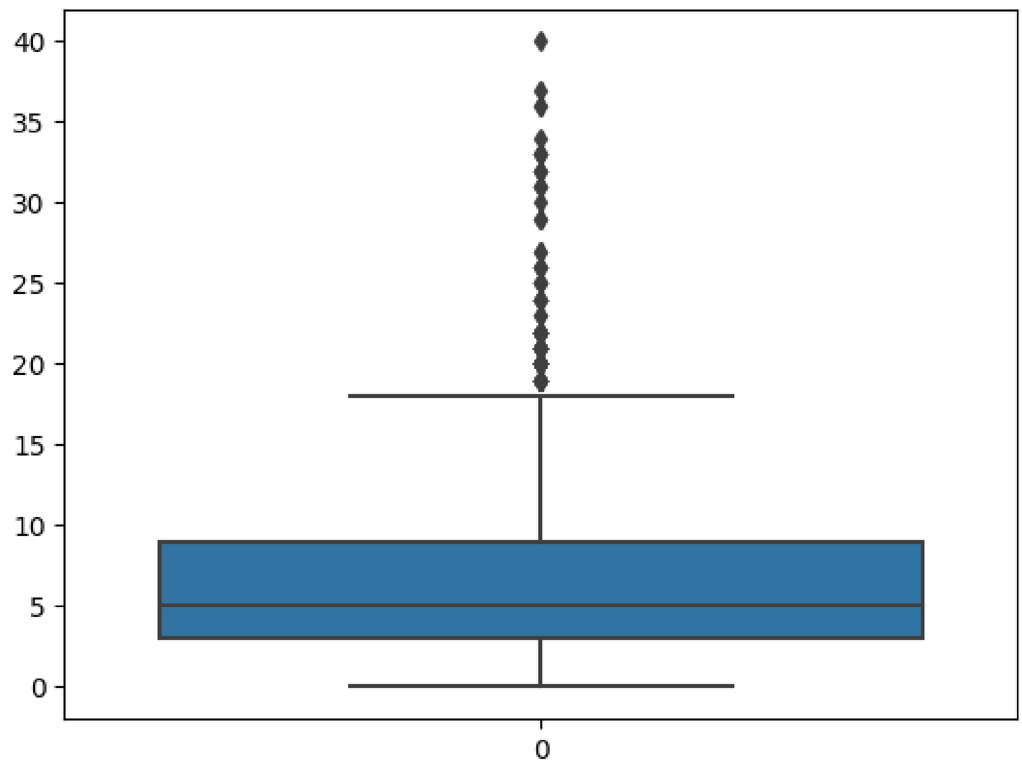
[29]: <Axes: >



- As of Now no outliers detected.

[30]:
```
sns.boxplot(df.YearsAtCompany)
```

[30]: <Axes: >



[31]:
```
Q1 = df.YearsAtCompany.quantile(0.25)
Q1
```

[31]: 3.0

[32]:
```
Q3 = df.YearsAtCompany.quantile(0.75)
Q3
```

[32]: 9.0

[33]:
```
IQR = Q3 - Q1
IQR
```

[33]: 6.0

[34]:
```
upperLimit = Q3 + 1.5*IQR
lowerLimit = Q1 - 1.5 * IQR
```

[35]:
```
print("Upper Limit : ",upperLimit)
```

```
Upper Limit :  18.0
```

[36]:
```
forUpperLimit = df["YearsAtCompany"]>upperLimit
forLowerLimit = df["YearsAtCompany"]<lowerLimit
totalOutliers = forUpperLimit + forLowerLimit
print("Total Outliers are : ",totalOutliers.sum())
```

```
Total Outliers are :  104
```

[37]:
```
df.shape
```

[37]:
```
(1470, 32)
```

- As there are 104 , than removing lets replace with median its best

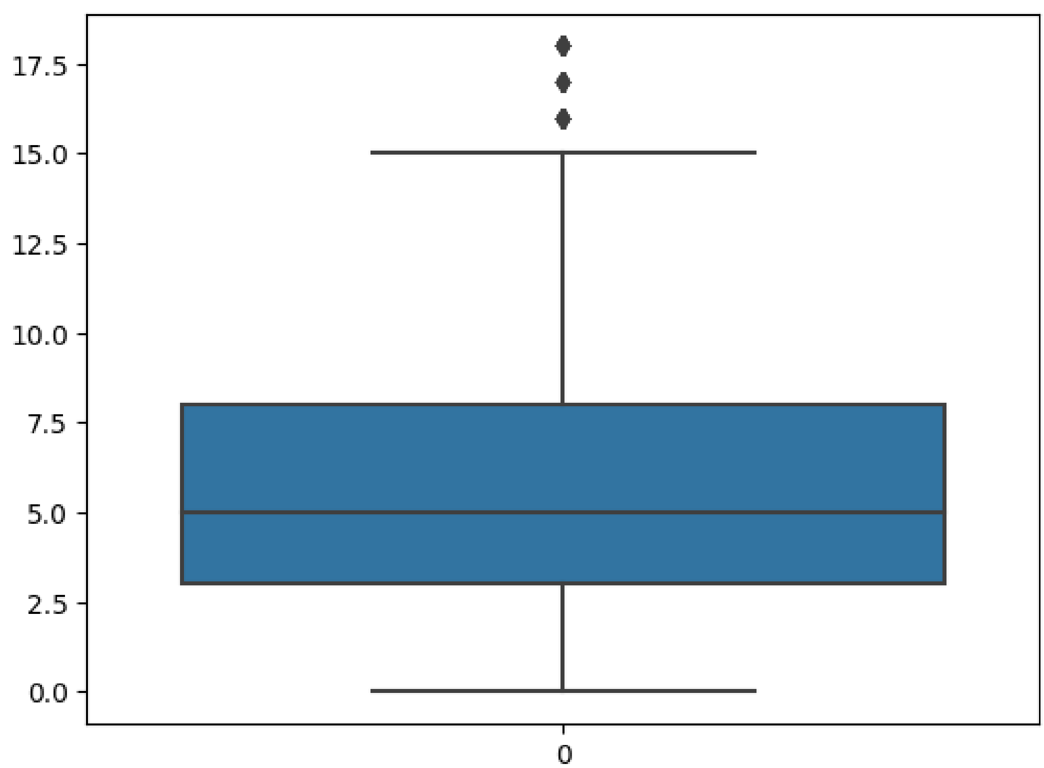
[38]:
```
x_YearsAtCompany = df["YearsAtCompany"].median()
x_YearsAtCompany
```

[38]:
```
5.0
```

[39]:
```
df["YearsAtCompany"] = np.where((df["YearsAtCompany"] > upperLimit) |␣
↪(df["YearsAtCompany"] < lowerLimit),x_YearsAtCompany, df["YearsAtCompany"])
```

[40]:
```
sns.boxplot(df.YearsAtCompany)
```

[40]:
```
<Axes: >
```



- Outliers Replace successfylly

[41]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 32 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   EnvironmentSatisfaction   1470 non-null   int64
 9   Gender                    1470 non-null   object
 10  HourlyRate                1470 non-null   int64
 11  JobInvolvement            1470 non-null   int64
 12  JobLevel                  1470 non-null   int64
 13  JobRole                   1470 non-null   object
 14  JobSatisfaction           1470 non-null   int64
 15  MaritalStatus             1470 non-null   object
 16  MonthlyIncome             1470 non-null   int64
 17  MonthlyRate               1470 non-null   int64
 18  NumCompaniesWorked        1470 non-null   int64
 19  OverTime                  1470 non-null   object
 20  PercentSalaryHike         1470 non-null   int64
 21  PerformanceRating         1470 non-null   int64
 22  RelationshipSatisfaction  1470 non-null   int64
 23  StockOptionLevel          1470 non-null   int64
 24  TotalWorkingYears         1470 non-null   int64
 25  TrainingTimesLastYear     1470 non-null   int64
 26  WorkLifeBalance           1470 non-null   int64
 27  YearsAtCompany            1470 non-null   float64
 28  YearsInCurrentRole        1470 non-null   int64
 29  YearsSinceLastPromotion   1470 non-null   int64
 30  YearsWithCurrManager      1470 non-null   int64
 31  Attrition_num             1470 non-null   int64
dtypes: float64(1), int64(23), object(8)
memory usage: 367.6+ KB
```

0.2.5 Splitting Dependent and Independent variables

[42]:
```
df.head()
```

[42]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  …  RelationshipSatisfaction  StockOptionLevel  TotalWorkingYears  \
0  Female  …                         1                 0                  8
1    Male  …                         4                 1                 10
2    Male  …                         2                 0                  7
3  Female  …                         3                 0                  8
4    Male  …                         4                 1                  6

  TrainingTimesLastYear  WorkLifeBalance YearsAtCompany  YearsInCurrentRole  \
0                     0                1            6.0                   4
1                     3                3           10.0                   7
2                     3                3            0.0                   0
3                     3                3            8.0                   7
4                     3                3            2.0                   2

   YearsSinceLastPromotion  YearsWithCurrManager Attrition_num
0                        0                     5             1
1                        1                     7             0
2                        0                     0             1
3                        3                     0             0
4                        2                     2             0

[5 rows x 32 columns]
```

[43]:
```
dependent = df["Attrition_num"]
```

[44]:
```
dependent
```

[44]:
```
0       1
1       0
```

```
2       1
3       0
4       0
       ..
1465    0
1466    0
1467    0
1468    0
1469    0
Name: Attrition_num, Length: 1470, dtype: int64
```

[45]:
```
independent = df.drop(columns=["Attrition_num", "Attrition"])
```

[46]:
```
independent.head()
```

[46]:
```
   Age     BusinessTravel  DailyRate              Department  \
0   41      Travel_Rarely       1102                   Sales
1   49  Travel_Frequently        279  Research & Development
2   37      Travel_Rarely       1373  Research & Development
3   33  Travel_Frequently       1392  Research & Development
4   27      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  HourlyRate  …  PerformanceRating  RelationshipSatisfaction  \
0  Female          94  …                  3                         1
1    Male          61  …                  4                         4
2    Male          92  …                  3                         2
3  Female          56  …                  3                         3
4    Male          40  …                  3                         4

  StockOptionLevel  TotalWorkingYears TrainingTimesLastYear  WorkLifeBalance  \
0                0                  8                     0                1
1                1                 10                     3                3
2                0                  7                     3                3
3                0                  8                     3                3
4                1                  6                     3                3

   YearsAtCompany  YearsInCurrentRole YearsSinceLastPromotion  \
0             6.0                   4                       0
1            10.0                   7                       1
2             0.0                   0                       0
```

```
3           8.0                    7                        3
4           2.0                    2                        2

   YearsWithCurrManager
0                     5
1                     7
2                     0
3                     0
4                     2

[5 rows x 30 columns]
```

- We have to make sure that independent is in DataFrame and dependent in Series

[47]:
```
type(independent)
```

[47]:
```
pandas.core.frame.DataFrame
```

[48]:
```
type(dependent)
```

[48]:
```
pandas.core.series.Series
```

0.2.6 Perform Encoding

[49]:
```
independent.head()
```

[49]:
```
   Age     BusinessTravel  DailyRate              Department  \
0   41      Travel_Rarely       1102                   Sales
1   49  Travel_Frequently        279  Research & Development
2   37      Travel_Rarely       1373  Research & Development
3   33  Travel_Frequently       1392  Research & Development
4   27      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  HourlyRate  …  PerformanceRating  RelationshipSatisfaction  \
0  Female          94  …                  3                         1
1    Male          61  …                  4                         4
2    Male          92  …                  3                         2
3  Female          56  …                  3                         3
4    Male          40  …                  3                         4
```

```
   StockOptionLevel  TotalWorkingYears TrainingTimesLastYear  WorkLifeBalance  \
0                 0                  8                     0                1
1                 1                 10                     3                3
2                 0                  7                     3                3
3                 0                  8                     3                3
4                 1                  6                     3                3

   YearsAtCompany  YearsInCurrentRole YearsSinceLastPromotion  \
0             6.0                   4                       0
1            10.0                   7                       1
2             0.0                   0                       0
3             8.0                   7                       3
4             2.0                   2                       2

   YearsWithCurrManager
0                     5
1                     7
2                     0
3                     0
4                     2

[5 rows x 30 columns]
```

[50]:
```
from sklearn.preprocessing import LabelEncoder
```

[51]:
```
le = LabelEncoder()
```

[52]:
```
independent["BusinessTravel"].value_counts()
```

[52]:
```
Travel_Rarely        1043
Travel_Frequently     277
Non-Travel            150
Name: BusinessTravel, dtype: int64
```

[53]:
```
independent["Department"].value_counts()
```

[53]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

[54]:
```
independent["EducationField"].value_counts()
```

[54]:
```
Life Sciences       606
Medical             464
Marketing           159
Technical Degree    132
```

```
Other                  82
Human Resources        27
Name: EducationField, dtype: int64
```

[55]:
```
independent["BusinessTravel"] = le.fit_transform(independent["BusinessTravel"])
```

[56]:
```
independent["BusinessTravel"].head()
```

[56]:
```
0    2
1    1
2    2
3    1
4    2
Name: BusinessTravel, dtype: int64
```

[57]:
```
independent["BusinessTravel"].tail()
```

[57]:
```
1465    1
1466    2
1467    2
1468    1
1469    2
Name: BusinessTravel, dtype: int64
```

[58]:
```
print(le.classes_)
```

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

[59]:
```
independent["Department"] = le.fit_transform(independent["Department"])
```

[60]:
```
independent["Department"].head()
```

[60]:
```
0    2
1    1
2    1
3    1
4    1
Name: Department, dtype: int64
```

[61]:
```
print(le.classes_)
```

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

[62]:
```
independent["EducationField"] = le.fit_transform(independent["EducationField"])
```

[63]:
```
independent["EducationField"].head()
```

[63]:
```
0    1
1    1
2    4
3    1
4    3
Name: EducationField, dtype: int64
```

[64]:
```
print(le.classes_)
```

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

[65]:
```
independent.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 30 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1470 non-null   int64
 1   BusinessTravel            1470 non-null   int64
 2   DailyRate                 1470 non-null   int64
 3   Department                1470 non-null   int64
 4   DistanceFromHome          1470 non-null   int64
 5   Education                 1470 non-null   int64
 6   EducationField            1470 non-null   int64
 7   EnvironmentSatisfaction   1470 non-null   int64
 8   Gender                    1470 non-null   object
 9   HourlyRate                1470 non-null   int64
 10  JobInvolvement            1470 non-null   int64
 11  JobLevel                  1470 non-null   int64
 12  JobRole                   1470 non-null   object
 13  JobSatisfaction           1470 non-null   int64
 14  MaritalStatus             1470 non-null   object
 15  MonthlyIncome             1470 non-null   int64
 16  MonthlyRate               1470 non-null   int64
 17  NumCompaniesWorked        1470 non-null   int64
 18  OverTime                  1470 non-null   object
 19  PercentSalaryHike         1470 non-null   int64
 20  PerformanceRating         1470 non-null   int64
 21  RelationshipSatisfaction  1470 non-null   int64
 22  StockOptionLevel          1470 non-null   int64
 23  TotalWorkingYears         1470 non-null   int64
 24  TrainingTimesLastYear     1470 non-null   int64
 25  WorkLifeBalance           1470 non-null   int64
 26  YearsAtCompany            1470 non-null   float64
 27  YearsInCurrentRole        1470 non-null   int64
 28  YearsSinceLastPromotion   1470 non-null   int64
```

```
 29  YearsWithCurrManager      1470 non-null   int64
dtypes: float64(1), int64(25), object(4)
memory usage: 344.7+ KB
```

- As there are still 5 so converting one by one better use for Loop

[66]:
```python
obj_col = independent.select_dtypes(include=['object']).columns


for column in obj_col:
    independent[column] = le.fit_transform(independent[column])
```

[67]:
```python
independent.head()
```

[67]:
```
   Age  BusinessTravel  DailyRate  Department  DistanceFromHome  Education  \
0   41               2       1102           2                 1          2
1   49               1        279           1                 8          1
2   37               2       1373           1                 2          2
3   33               1       1392           1                 3          4
4   27               2        591           1                 2          1

   EducationField  EnvironmentSatisfaction  Gender  HourlyRate  …  \
0               1                        2       0          94  …
1               1                        3       1          61  …
2               4                        4       1          92  …
3               1                        4       0          56  …
4               3                        1       1          40  …

   PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
0                  3                         1                 0
1                  4                         4                 1
2                  3                         2                 0
3                  3                         3                 0
4                  3                         4                 1

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0                  8                      0                1             6.0
1                 10                      3                3            10.0
2                  7                      3                3             0.0
3                  8                      3                3             8.0
4                  6                      3                3             2.0

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                   4                        0                     5
1                   7                        1                     7
2                   0                        0                     0
3                   7                        3                     0
4                   2                        2                     2
```

```
[5 rows x 30 columns]
```

[68]:
```python
independent.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 30 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Age                       1470 non-null   int64  
 1   BusinessTravel            1470 non-null   int64  
 2   DailyRate                 1470 non-null   int64  
 3   Department                1470 non-null   int64  
 4   DistanceFromHome          1470 non-null   int64  
 5   Education                 1470 non-null   int64  
 6   EducationField            1470 non-null   int64  
 7   EnvironmentSatisfaction   1470 non-null   int64  
 8   Gender                    1470 non-null   int64  
 9   HourlyRate                1470 non-null   int64  
 10  JobInvolvement            1470 non-null   int64  
 11  JobLevel                  1470 non-null   int64  
 12  JobRole                   1470 non-null   int64  
 13  JobSatisfaction           1470 non-null   int64  
 14  MaritalStatus             1470 non-null   int64  
 15  MonthlyIncome             1470 non-null   int64  
 16  MonthlyRate               1470 non-null   int64  
 17  NumCompaniesWorked        1470 non-null   int64  
 18  OverTime                  1470 non-null   int64  
 19  PercentSalaryHike         1470 non-null   int64  
 20  PerformanceRating         1470 non-null   int64  
 21  RelationshipSatisfaction  1470 non-null   int64  
 22  StockOptionLevel          1470 non-null   int64  
 23  TotalWorkingYears         1470 non-null   int64  
 24  TrainingTimesLastYear     1470 non-null   int64  
 25  WorkLifeBalance           1470 non-null   int64  
 26  YearsAtCompany            1470 non-null   float64
 27  YearsInCurrentRole        1470 non-null   int64  
 28  YearsSinceLastPromotion   1470 non-null   int64  
 29  YearsWithCurrManager      1470 non-null   int64  
dtypes: float64(1), int64(29)
memory usage: 344.7 KB
```

- All are in int64 so Perfect!

0.2.7 Splitting Data into Train and Test

[69]:
```
from sklearn.model_selection import train_test_split as tts
```

[70]:
```
independent.shape, dependent.shape
```

[70]:
```
((1470, 30), (1470,))
```

[71]:
```
independent_train,independent_test,dependent_train,dependent_test =␣
↪tts(independent,dependent,test_size=0.2,random_state=0)
```

[72]:
```
independent_train.shape , independent_test.shape, dependent_train.shape,␣
↪dependent_test.shape
```

[72]:
```
((1176, 30), (294, 30), (1176,), (294,))
```

Below are train and test without feature scaling.

[73]:
```
independent_wfs_train = independent_train
dependent_wfs_train = dependent_train

independent_wfs_test = independent_test
dependent_wfs_test = dependent_test
```

0.2.8 Feature Scaling

- Only for independent we will perform Feature Scaling

[74]:
```
from sklearn.preprocessing import MinMaxScaler

ms = MinMaxScaler()
```

[75]:
```
independent_train.columns
```

[75]:
```
Index(['Age', 'BusinessTravel', 'DailyRate', 'Department', 'DistanceFromHome',
       'Education', 'EducationField', 'EnvironmentSatisfaction', 'Gender',
       'HourlyRate', 'JobInvolvement', 'JobLevel', 'JobRole',
       'JobSatisfaction', 'MaritalStatus', 'MonthlyIncome', 'MonthlyRate',
       'NumCompaniesWorked', 'OverTime', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

[76]:
```
independent_train=pd.DataFrame(ms.
↪fit_transform(independent_train),columns=independent_train.columns)
independent_train.head()
```

[76]:
```
        Age  BusinessTravel  DailyRate  Department  DistanceFromHome  \
0  0.952381             1.0   0.359140         1.0          0.714286
1  0.642857             1.0   0.606452         0.5          0.964286
2  0.523810             1.0   0.140502         1.0          0.892857
3  0.428571             0.0   0.953405         1.0          0.250000
4  0.166667             0.5   0.354839         1.0          0.821429

   Education  EducationField  EnvironmentSatisfaction  Gender  HourlyRate  \
0       0.50             0.2                 1.000000     0.0    0.600000
1       0.50             1.0                 1.000000     1.0    0.957143
2       0.50             0.4                 0.666667     1.0    0.628571
3       0.75             0.2                 0.000000     1.0    0.657143
4       0.00             0.2                 0.666667     1.0    0.614286

   …  PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
0  …                0.0                  0.666667          0.333333
1  …                1.0                  1.000000          0.333333
2  …                0.0                  0.333333          0.333333
3  …                0.0                  0.333333          0.000000
4  …                0.0                  1.000000          0.000000

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0              0.725               0.333333         0.333333        0.055556
1              0.200               0.500000         0.666667        0.277778
2              0.200               0.500000         0.333333        0.388889
3              0.250               0.166667         0.666667        0.555556
4              0.025               0.666667         0.666667        0.055556

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0            0.000000                 0.000000              0.000000
1            0.222222                 0.000000              0.176471
2            0.388889                 0.466667              0.294118
3            0.388889                 0.000000              0.529412
4            0.000000                 0.066667              0.000000

[5 rows x 30 columns]
```

[77]:
```
independent_test=pd.DataFrame(ms.
 ↪fit_transform(independent_test),columns=independent_test.columns)
independent_test.head()
```

[77]:
```
        Age  BusinessTravel  DailyRate  Department  DistanceFromHome  \
0  0.428571             0.0   0.382353         1.0          0.321429
1  0.357143             1.0   0.339311         0.5          0.857143
2  0.404762             0.5   0.401722         1.0          0.607143
3  0.523810             1.0   0.997131         0.5          0.678571
4  0.261905             0.5   0.256098         0.5          0.821429
```

```
   Education  EducationField  EnvironmentSatisfaction  Gender  HourlyRate  \
0       0.75             0.6                 0.333333     1.0    0.028571
1       0.50             0.2                 1.000000     1.0    0.200000
2       0.75             0.4                 1.000000     0.0    0.528571
3       0.75             1.0                 0.000000     1.0    0.442857
4       0.25             0.2                 1.000000     1.0    0.614286

   …  PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
0  …                0.0                  1.000000          0.000000
1  …                0.0                  1.000000          0.000000
2  …                0.0                  0.666667          0.333333
3  …                1.0                  1.000000          0.333333
4  …                1.0                  0.333333          0.000000

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0           0.270270               0.500000         0.333333        0.555556
1           0.135135               0.333333         0.666667        0.277778
2           0.135135               0.000000         0.333333        0.222222
3           0.378378               1.000000         0.666667        0.611111
4           0.027027               0.500000         0.333333        0.055556

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0            0.176471                 0.600000              0.411765
1            0.176471                 0.000000              0.117647
2            0.117647                 0.200000              0.117647
3            0.588235                 0.733333              0.058824
4            0.000000                 0.066667              0.000000

[5 rows x 30 columns]
```

0.2.9 Model Building Using Logistic Regression

Import the model building Libraries

[78]:
```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
```

Initializing the model

[79]:
```
model.fit(independent_train,dependent_train)
```

[79]:
```
LogisticRegression()
```

[80]:
```
type(dependent_test)
```

[80]:
```
pandas.core.series.Series
```

Training and testing the model

[81]:
```
pred=model.predict(independent_test)
pred
```

[81]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 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, 1, 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, 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,
       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, 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, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 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, 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, 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, 1, 0, 0])
```

[82]:
```
dependent_test
```

[82]:
```
442     0
1091    0
981     1
785     0
1332    1
       ..
1439    0
481     0
124     1
198     0
1229    0
Name: Attrition_num, Length: 294, dtype: int64
```

[83]:
```
dfActPred = pd.DataFrame({"Actual":dependent_test,"Predicted":pred})
```

[84]:
```
dfActPred.head()
```

[84]:

	Actual	Predicted
442	0	0
1091	0	0
981	1	0
785	0	0
1332	1	1

[85]:
```
dfActPred.tail()
```

[85]:
```
      Actual  Predicted
1439       0          0
481        0          0
124        1          1
198        0          0
1229       0          0
```

0.2.10 Evaluation of classification model

- Accuracy Score

[86]:
```
from sklearn.metrics import␣
↪accuracy_score,confusion_matrix,classification_report,roc_auc_score,roc_curve
```

[87]:
```
accuracy_score(dependent_test,pred)
```

[87]:
```
0.8775510204081632
```

[88]:
```
confusion_matrix(dependent_test,pred)
```

[88]:
```
array([[241,   4],
       [ 32,  17]])
```

[89]:
```
pd.crosstab(dependent_test,pred)
```

[89]:
```
col_0            0   1
Attrition_num
0              241   4
1               32  17
```

TN - 242

FP - 3

FN - 31

TP - 18

[90]:
```
(242+18)/(242+3+31+18)
```

[90]:
```
0.8843537414965986
```

- Support-> Gives actual values , actually 245(0's) and 49(1's) are there

[91]:
```
print(classification_report(dependent_test,pred))
```

```
              precision    recall  f1-score   support

           0       0.88      0.98      0.93       245
           1       0.81      0.35      0.49        49

    accuracy                           0.88       294
   macro avg       0.85      0.67      0.71       294
weighted avg       0.87      0.88      0.86       294
```

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F1 Score = 2 x (Precision x Recall) / (Precision + Recall)

[92]:
```
precision_formula = (18)/(18+3)
recall_formula = (18)/(18+31)

precision_formula, recall_formula
```

[92]: (0.8571428571428571, 0.3673469387755102)

[93]:
```
f1_score_formula = 2*(precision_formula*recall_formula)/
 ↪(precision_formula+recall_formula)
f1_score_formula
```

[93]: 0.5142857142857143

[94]:
```
from sklearn.metrics import precision_score, recall_score,f1_score
```

[95]:
```
precision = precision_score(dependent_test,pred)
recall = recall_score(dependent_test,pred)
```

[96]:
```
precision, recall
```

[96]: (0.8095238095238095, 0.3469387755102041)

[97]:
```
f1Score = f1_score(dependent_test,pred)
f1Score
```

[97]: 0.4857142857142857

0.2.11 ROC-AOC Curve

[98]:
```
probability=model.predict_proba(independent_test)[:,1]
```

[99]:
```
probability
```

```
[99]: array([0.12922371, 0.19907451, 0.32214642, 0.07282822, 0.67550013,
       0.06458206, 0.56993732, 0.06352598, 0.00480938, 0.36158894,
       0.05910439, 0.3178773 , 0.0190863 , 0.68299344, 0.21568267,
       0.03282269, 0.09937107, 0.17884424, 0.05044214, 0.20783213,
       0.25464102, 0.01377538, 0.05775749, 0.05413135, 0.57047727,
       0.41277701, 0.06416197, 0.03537188, 0.71943722, 0.06332207,
       0.0142358 , 0.02945315, 0.07824106, 0.18161948, 0.07320006,
       0.02979669, 0.10066203, 0.07532229, 0.03191684, 0.05114037,
       0.08678858, 0.01865009, 0.01431941, 0.00975379, 0.02454072,
       0.52187579, 0.40791755, 0.00348931, 0.76721244, 0.49560054,
       0.1197844 , 0.47330723, 0.07081607, 0.25294683, 0.6934059 ,
       0.27224945, 0.02051497, 0.30360798, 0.02704844, 0.17727852,
       0.02223483, 0.23263883, 0.16029035, 0.03380835, 0.39941272,
       0.03767792, 0.25863197, 0.1288367 , 0.08927058, 0.10680752,
       0.07308503, 0.29719199, 0.07015979, 0.07487547, 0.11814906,
       0.06598593, 0.05367264, 0.07750627, 0.20781535, 0.03346305,
       0.00682096, 0.02450951, 0.14810222, 0.02709382, 0.03388168,
       0.07821652, 0.00721605, 0.03607966, 0.04032811, 0.14602371,
       0.31655041, 0.16396522, 0.28694302, 0.26391696, 0.01991403,
       0.19405465, 0.34266084, 0.27661631, 0.07477764, 0.04767292,
       0.2455894 , 0.73996242, 0.35809942, 0.01960873, 0.09562256,
       0.02828143, 0.05406036, 0.15716951, 0.05794308, 0.13129884,
       0.08033989, 0.05190153, 0.02539211, 0.14661551, 0.06150688,
       0.02995383, 0.04350927, 0.11417131, 0.00620497, 0.01244031,
       0.1543817 , 0.04979925, 0.06977358, 0.81227301, 0.02979278,
       0.02098512, 0.00908864, 0.13270146, 0.16084703, 0.05044345,
       0.01657526, 0.27713174, 0.55178948, 0.32969314, 0.03874518,
       0.41762468, 0.56503335, 0.14277993, 0.08476358, 0.27028741,
       0.10050483, 0.07027489, 0.11005442, 0.13168984, 0.19769234,
       0.02678849, 0.18408845, 0.00618421, 0.06581633, 0.15714696,
       0.05915219, 0.15589606, 0.06295905, 0.14709954, 0.03125142,
       0.02096167, 0.06699647, 0.07549296, 0.0143998 , 0.0102274 ,
       0.4865774 , 0.01045408, 0.1540103 , 0.82191668, 0.10439188,
       0.27459021, 0.16859167, 0.13377985, 0.0331703 , 0.0061185 ,
       0.03749531, 0.08002025, 0.12220619, 0.11135701, 0.02332668,
       0.14545474, 0.11315407, 0.08644624, 0.05206343, 0.10046823,
       0.02847359, 0.09763148, 0.00626403, 0.7910294 , 0.0401393 ,
       0.04236831, 0.38924552, 0.04458454, 0.72979454, 0.1222053 ,
       0.4026196 , 0.41690991, 0.29716733, 0.05272645, 0.07974829,
       0.15291533, 0.04184228, 0.01284835, 0.29000444, 0.05246719,
       0.14030675, 0.15726889, 0.68703415, 0.06380904, 0.23330852,
       0.03350802, 0.50295316, 0.0027797 , 0.13605052, 0.02473489,
       0.11862706, 0.17514936, 0.05336843, 0.10843099, 0.14568324,
       0.02609283, 0.02103642, 0.07300879, 0.03184676, 0.15389922,
       0.0913993 , 0.21604817, 0.75345085, 0.12962065, 0.3907942 ,
       0.01401912, 0.11495867, 0.24970044, 0.35434274, 0.04231722,
       0.039431  , 0.30909031, 0.05571662, 0.01656217, 0.16897215,
```

```
       0.37085108, 0.26602834, 0.00755729, 0.09047066, 0.00931448,
       0.14602227, 0.26794103, 0.01140113, 0.16914581, 0.03949658,
       0.03614227, 0.39205863, 0.37002035, 0.03745161, 0.11238001,
       0.3694244 , 0.31494318, 0.80605996, 0.04624853, 0.20543001,
       0.07243583, 0.00550765, 0.68824999, 0.38106453, 0.35993989,
       0.38055498, 0.0311644 , 0.19047036, 0.06154749, 0.06549142,
       0.10660427, 0.00736727, 0.23593521, 0.47591789, 0.07394317,
       0.08987746, 0.01261264, 0.14220915, 0.05432877, 0.02094427,
       0.02758973, 0.0617762 , 0.25318654, 0.25538526, 0.20155549,
       0.27325143, 0.01750714, 0.1580262 , 0.0832157 , 0.02755938,
       0.20392804, 0.00919492, 0.23655275, 0.00443645, 0.02352029,
       0.20844774, 0.72893829, 0.0740282 , 0.29540129])
```

[100]:
```
fpr,tpr,threshsholds = roc_curve(dependent_test,probability)
```

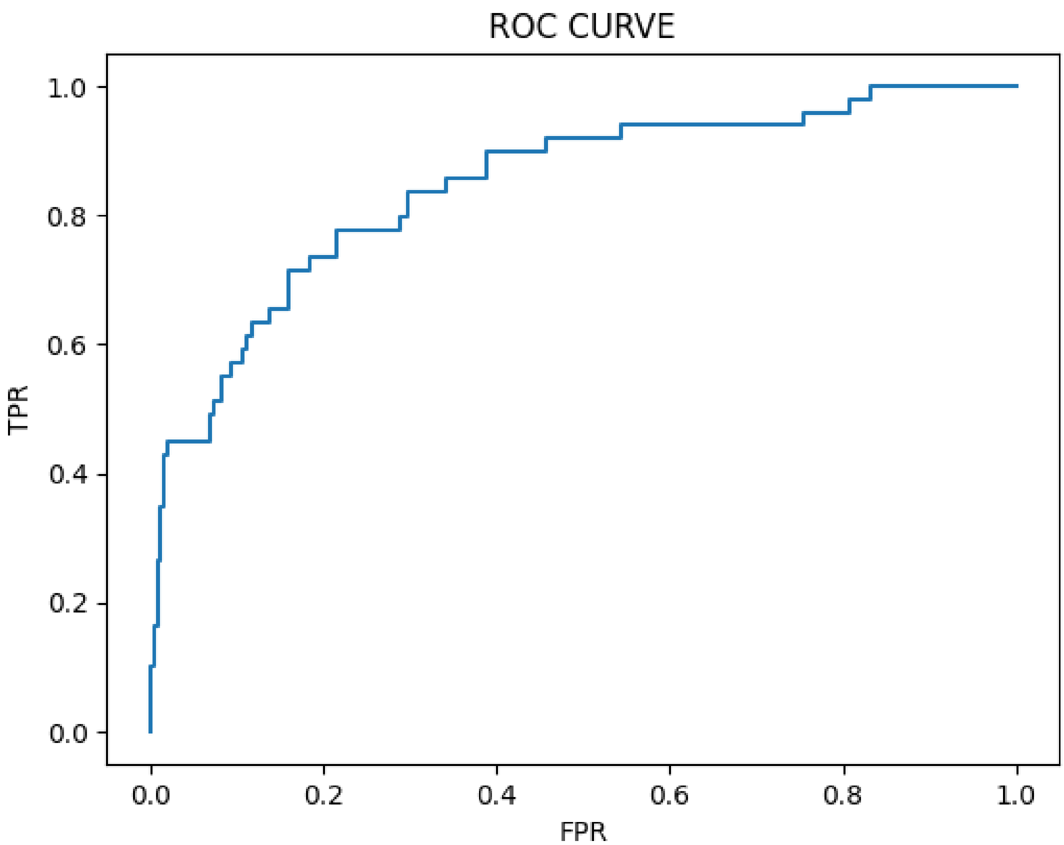
[101]:
```
fpr,tpr,threshsholds
```

[101]:
```
(array([0.        , 0.        , 0.        , 0.00408163, 0.00408163,
        0.00816327, 0.00816327, 0.0122449 , 0.0122449 , 0.01632653,
        0.01632653, 0.02040816, 0.02040816, 0.06938776, 0.06938776,
        0.07346939, 0.07346939, 0.08163265, 0.08163265, 0.09387755,
        0.09387755, 0.10612245, 0.10612245, 0.11020408, 0.11020408,
        0.11836735, 0.11836735, 0.13877551, 0.13877551, 0.15918367,
        0.15918367, 0.18367347, 0.18367347, 0.21632653, 0.21632653,
        0.28979592, 0.28979592, 0.29795918, 0.29795918, 0.34285714,
        0.34285714, 0.3877551 , 0.3877551 , 0.45714286, 0.45714286,
        0.54285714, 0.54285714, 0.75510204, 0.75510204, 0.80816327,
        0.80816327, 0.83265306, 0.83265306, 1.        ]),
 array([0.        , 0.02040816, 0.10204082, 0.10204082, 0.16326531,
        0.16326531, 0.26530612, 0.26530612, 0.34693878, 0.34693878,
        0.42857143, 0.42857143, 0.44897959, 0.44897959, 0.48979592,
        0.48979592, 0.51020408, 0.51020408, 0.55102041, 0.55102041,
        0.57142857, 0.57142857, 0.59183673, 0.59183673, 0.6122449 ,
        0.6122449 , 0.63265306, 0.63265306, 0.65306122, 0.65306122,
        0.71428571, 0.71428571, 0.73469388, 0.73469388, 0.7755102 ,
        0.7755102 , 0.79591837, 0.79591837, 0.83673469, 0.83673469,
        0.85714286, 0.85714286, 0.89795918, 0.89795918, 0.91836735,
        0.91836735, 0.93877551, 0.93877551, 0.95918367, 0.95918367,
        0.97959184, 0.97959184, 1.        , 1.        ]),
 array([1.82191668, 0.82191668, 0.76721244, 0.75345085, 0.72893829,
        0.71943722, 0.67550013, 0.57047727, 0.52187579, 0.50295316,
        0.47330723, 0.41762468, 0.41690991, 0.3694244 , 0.35993989,
        0.35809942, 0.35434274, 0.32969314, 0.3178773 , 0.30909031,
        0.30360798, 0.29540129, 0.29000444, 0.28694302, 0.27713174,
        0.27459021, 0.27325143, 0.26391696, 0.25863197, 0.24970044,
        0.23593521, 0.20783213, 0.20781535, 0.18408845, 0.17884424,
        0.14810222, 0.14709954, 0.14602371, 0.14568324, 0.12922371,
```

```
0.1288367 , 0.11005442, 0.10680752, 0.08033989, 0.08002025,
0.06598593, 0.06581633, 0.03282269, 0.03191684, 0.02704844,
0.02678849, 0.02352029, 0.02332668, 0.0027797 ]))
```

- Area under this curve is AUC

[102]:
```
plt.plot(fpr,tpr)
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC CURVE')
plt.show()
```



0.2.12 Model Building Using Decision Tree

[103]:
```
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier()
```

[104]:
```
dtc.fit(independent_wfs_train,dependent_wfs_train)
```

[104]:
```
DecisionTreeClassifier()
```

[105]:
```
pred_dtc=dtc.predict(independent_wfs_test)
pred_dtc
```

[105]:
```
array([0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 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,
       1, 0, 1, 0, 0, 1, 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, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0])
```

0.2.13 Evaluation

[106]:
```
accuracy_score(dependent_wfs_test,pred_dtc)
```

[106]:
```
0.7755102040816326
```

0.2.14 Tree

[107]:
```
from sklearn import tree
```

[108]:
```
import graphviz
```

[109]:
```
from sklearn import tree
import graphviz

dot_data = tree.export_graphviz(dtc,
 out_file=None,feature_names=None,class_names=None,filled=True)

graph = graphviz.Source(dot_data)
graph.format = 'png'
graph.render('dtree_render', view=True, cleanup=True)
```

[109]:
```
'dtree_render.png'
```

0.2.15 Hyper Parameters Tuning

- To increase the accuracy we use Hyper Parameter tuning

[110]:
```
from sklearn.model_selection import GridSearchCV
```

Pre-Pruning

[111]:
```
parameter_dtc = {
 'criterion': ['gini', 'entropy'],
 'splitter': ['best', 'random'],
 'max_depth': [None, 3, 5, 10, 15, 20, 25, 30],
 'min_samples_split': [2, 5, 10, 20, 30],
 'min_samples_leaf': [1, 2, 5, 10],
 'max_features': ['auto', 'sqrt', 'log2', None],
 'class_weight': [None, 'balanced'],
 'max_leaf_nodes': [None, 10, 20, 30, 40],
}
```

[112]:
```
gridSearch=GridSearchCV(estimator=dtc,param_grid=parameter_dtc,cv=5,scoring='accuracy')
```

[113]:
```
gridSearch.fit(independent_wfs_train, dependent_wfs_train)
```

[113]:
```
GridSearchCV(cv=5, estimator=DecisionTreeClassifier(),
             param_grid={'class_weight': [None, 'balanced'],
                         'criterion': ['gini', 'entropy'],
                         'max_depth': [None, 3, 5, 10, 15, 20, 25, 30],
                         'max_features': ['auto', 'sqrt', 'log2', None],
                         'max_leaf_nodes': [None, 10, 20, 30, 40],
                         'min_samples_leaf': [1, 2, 5, 10],
                         'min_samples_split': [2, 5, 10, 20, 30],
                         'splitter': ['best', 'random']},
             scoring='accuracy')
```

[114]:
```
best_params_gs = gridSearch.best_params_
best_params_gs
```

[114]:
```
{'class_weight': None,
 'criterion': 'entropy',
 'max_depth': 20,
 'max_features': None,
 'max_leaf_nodes': None,
 'min_samples_leaf': 10,
 'min_samples_split': 30,
 'splitter': 'random'}
```

[115]:
```
dtc_cv = DecisionTreeClassifier(**best_params_gs)
```

[116]:
```
dtc_cv.fit(independent_wfs_train,dependent_wfs_train)
```

[116]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=20, min_samples_leaf=10,
                       min_samples_split=30, splitter='random')
```

[117]:
```
pred_dtc_cv = dtc_cv.predict(independent_wfs_test)
```

[118]:
```
print(classification_report(dependent_wfs_test,pred_dtc_cv))
```

```
              precision    recall  f1-score   support

           0       0.87      0.91      0.89       245
           1       0.42      0.31      0.35        49

    accuracy                           0.81       294
   macro avg       0.64      0.61      0.62       294
weighted avg       0.79      0.81      0.80       294
```

- Got 81% accuracy which is Excellent