

# 21BCE8975 - Assignment-4

September 27, 2023

## 1 Assignment-4

Name: Chigulla Somu Vinay Raj

Registration number: 21BCE8975

Email: vinayraj.21bce8975@vitapstudent.ac.in

Mobile Number: 9491604860

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

```
[2]: data=pd.read_csv("Employee-Attrition.csv")
```

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

```
[3]:   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   EmployeeCount   EmployeeNumber \
0              1          2   Life Sciences              1              1
1              8          1   Life Sciences              1              2
2              2          2          Other              1              4
3              3          4   Life Sciences              1              5
4              2          1          Medical              1              7

   ... RelationshipSatisfaction   StandardHours   StockOptionLevel \
0   ...              1              80              0
1   ...              4              80              1
2   ...              2              80              0
3   ...              3              80              0
4   ...              4              80              1
```

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

	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 35 columns]

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

```
[4]:
```

	Age	Attrition	BusinessTravel	DailyRate	Department	\
1465	36	No	Travel_Frequently	884	Research & Development	
1466	39	No	Travel_Rarely	613	Research & Development	
1467	27	No	Travel_Rarely	155	Research & Development	
1468	49	No	Travel_Frequently	1023	Sales	
1469	34	No	Travel_Rarely	628	Research & Development	

	DistanceFromHome	Education	EducationField	EmployeeCount	\
1465	23	2	Medical	1	
1466	6	1	Medical	1	
1467	4	3	Life Sciences	1	
1468	2	3	Medical	1	
1469	8	3	Medical	1	

	EmployeeNumber	...	RelationshipSatisfaction	StandardHours	\
1465	2061	...	3	80	
1466	2062	...	1	80	
1467	2064	...	2	80	
1468	2065	...	4	80	
1469	2068	...	1	80	

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	\
1465	1	17	3	
1466	1	9	5	
1467	1	6	0	
1468	0	17	3	
1469	0	6	3	

	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole	\
1465	3	5	2	
1466	3	7	7	
1467	3	6	2	
1468	2	9	6	
1469	4	4	3	

	YearsSinceLastPromotion	YearsWithCurrManager
1465	0	3
1466	1	7
1467	0	3
1468	0	8
1469	1	2

[5 rows x 35 columns]

```
[5]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Age                                    1470 non-null   int64
1   Attrition                            1470 non-null   object
2   BusinessTravel                       1470 non-null   object
3   DailyRate                            1470 non-null   int64
4   Department                           1470 non-null   object
5   DistanceFromHome                     1470 non-null   int64
6   Education                             1470 non-null   int64
7   EducationField                       1470 non-null   object
8   EmployeeCount                        1470 non-null   int64
9   EmployeeNumber                       1470 non-null   int64
10  EnvironmentSatisfaction               1470 non-null   int64
11  Gender                               1470 non-null   object
12  HourlyRate                           1470 non-null   int64
13  JobInvolvement                       1470 non-null   int64
14  JobLevel                             1470 non-null   int64
15  JobRole                              1470 non-null   object
16  JobSatisfaction                      1470 non-null   int64
17  MaritalStatus                        1470 non-null   object
18  MonthlyIncome                       1470 non-null   int64
19  MonthlyRate                          1470 non-null   int64
20  NumCompaniesWorked                   1470 non-null   int64
21  Over18                              1470 non-null   object
22  OverTime                             1470 non-null   object
23  PercentSalaryHike                    1470 non-null   int64
24  PerformanceRating                    1470 non-null   int64
```

```

25 RelationshipSatisfaction 1470 non-null int64
26 StandardHours           1470 non-null int64
27 StockOptionLevel        1470 non-null int64
28 TotalWorkingYears       1470 non-null int64
29 TrainingTimesLastYear   1470 non-null int64
30 WorkLifeBalance         1470 non-null int64
31 YearsAtCompany          1470 non-null int64
32 YearsInCurrentRole      1470 non-null int64
33 YearsSinceLastPromotion 1470 non-null int64
34 YearsWithCurrManager    1470 non-null int64

```

dtypes: int64(26), object(9)

memory usage: 402.1+ KB

```
[6]: data.describe()
```

```

[6]:      Age      DailyRate  DistanceFromHome  Education  EmployeeCount  \
count  1470.000000  1470.000000      1470.000000  1470.000000      1470.0
mean    36.923810   802.485714         9.192517    2.912925         1.0
std      9.135373   403.509100         8.106864    1.024165         0.0
min     18.000000   102.000000         1.000000    1.000000         1.0
25%     30.000000   465.000000         2.000000    2.000000         1.0
50%     36.000000   802.000000         7.000000    3.000000         1.0
75%     43.000000  1157.000000        14.000000    4.000000         1.0
max     60.000000  1499.000000        29.000000    5.000000         1.0

```

```

      EmployeeNumber  EnvironmentSatisfaction  HourlyRate  JobInvolvement  \
count      1470.000000      1470.000000  1470.000000  1470.000000
mean     1024.865306         2.721769    65.891156    2.729932
std       602.024335         1.093082   20.329428    0.711561
min         1.000000         1.000000   30.000000    1.000000
25%       491.250000         2.000000   48.000000    2.000000
50%      1020.500000         3.000000   66.000000    3.000000
75%      1555.750000         4.000000   83.750000    3.000000
max      2068.000000         4.000000  100.000000    4.000000

```

```

      JobLevel  ...  RelationshipSatisfaction  StandardHours  \
count  1470.000000  ...      1470.000000      1470.0
mean     2.063946  ...         2.712245         80.0
std      1.106940  ...         1.081209         0.0
min      1.000000  ...         1.000000         80.0
25%      1.000000  ...         2.000000         80.0
50%      2.000000  ...         3.000000         80.0
75%      3.000000  ...         4.000000         80.0
max      5.000000  ...         4.000000         80.0

```

```

      StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
count      1470.000000      1470.000000      1470.000000

```

mean	0.793878	11.279592	2.799320
std	0.852077	7.780782	1.289271
min	0.000000	0.000000	0.000000
25%	0.000000	6.000000	2.000000
50%	1.000000	10.000000	3.000000
75%	1.000000	15.000000	3.000000
max	3.000000	40.000000	6.000000

	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole \
count	1470.000000	1470.000000	1470.000000
mean	2.761224	7.008163	4.229252
std	0.706476	6.126525	3.623137
min	1.000000	0.000000	0.000000
25%	2.000000	3.000000	2.000000
50%	3.000000	5.000000	3.000000
75%	3.000000	9.000000	7.000000
max	4.000000	40.000000	18.000000

	YearsSinceLastPromotion	YearsWithCurrManager
count	1470.000000	1470.000000
mean	2.187755	4.123129
std	3.222430	3.568136
min	0.000000	0.000000
25%	0.000000	2.000000
50%	1.000000	3.000000
75%	3.000000	7.000000
max	15.000000	17.000000

[8 rows x 26 columns]

## 1.1 Handling Null Values

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

```
[7]: Age                False
Attrition              False
BusinessTravel         False
DailyRate              False
Department             False
DistanceFromHome       False
Education              False
EducationField          False
EmployeeCount          False
EmployeeNumber         False
EnvironmentSatisfaction False
Gender                 False
HourlyRate             False
```

JobInvolvement	False
JobLevel	False
JobRole	False
JobSatisfaction	False
MaritalStatus	False
MonthlyIncome	False
MonthlyRate	False
NumCompaniesWorked	False
Over18	False
OverTime	False
PercentSalaryHike	False
PerformanceRating	False
RelationshipSatisfaction	False
StandardHours	False
StockOptionLevel	False
TotalWorkingYears	False
TrainingTimesLastYear	False
WorkLifeBalance	False
YearsAtCompany	False
YearsInCurrentRole	False
YearsSinceLastPromotion	False
YearsWithCurrManager	False

dtype: bool

```
[8]: data.isnull().sum()
```

```
[8]: 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
```

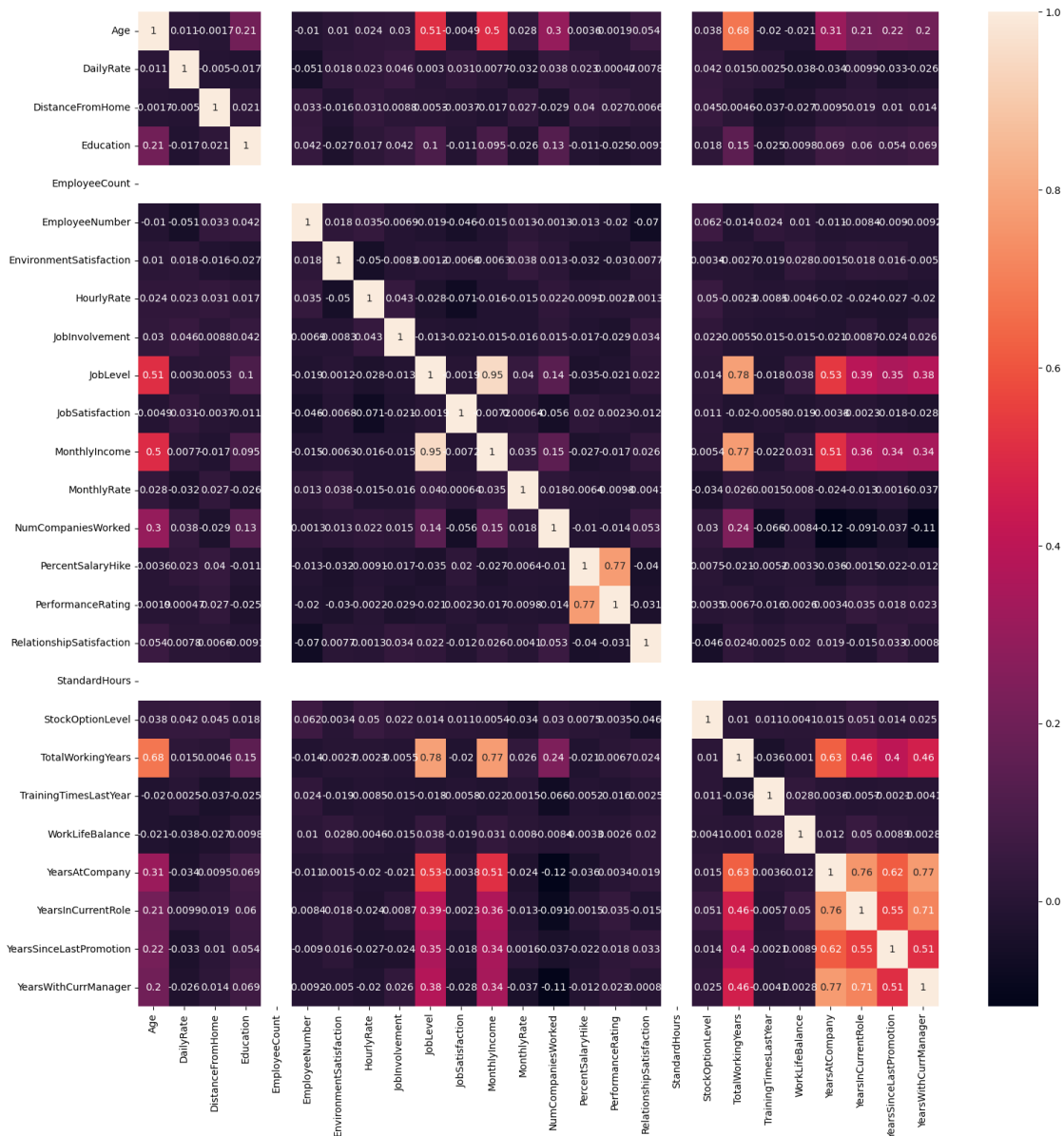
```
[9]: cor=data.corr()
```

```
C:\Users\MSI\AppData\Local\Temp\ipykernel_9064\1426905697.py:1: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future
version, it will default to False. Select only valid columns or specify the
value of numeric_only to silence this warning.
```

```
cor=data.corr()
```

```
[10]: fig=plt.figure(figsize=(18,18))
      sns.heatmap(cor,annot=True)
```

```
[10]: <Axes: >
```

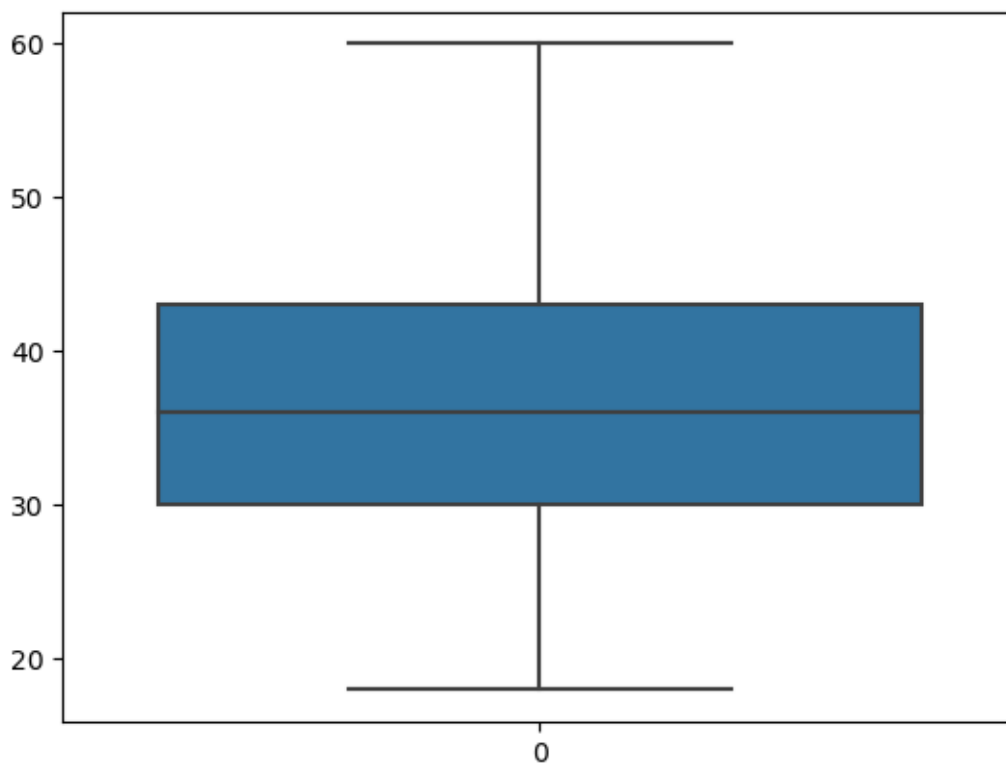


## 1.2 Outliers

```
[11]: sns.boxplot(data["Age"])
```

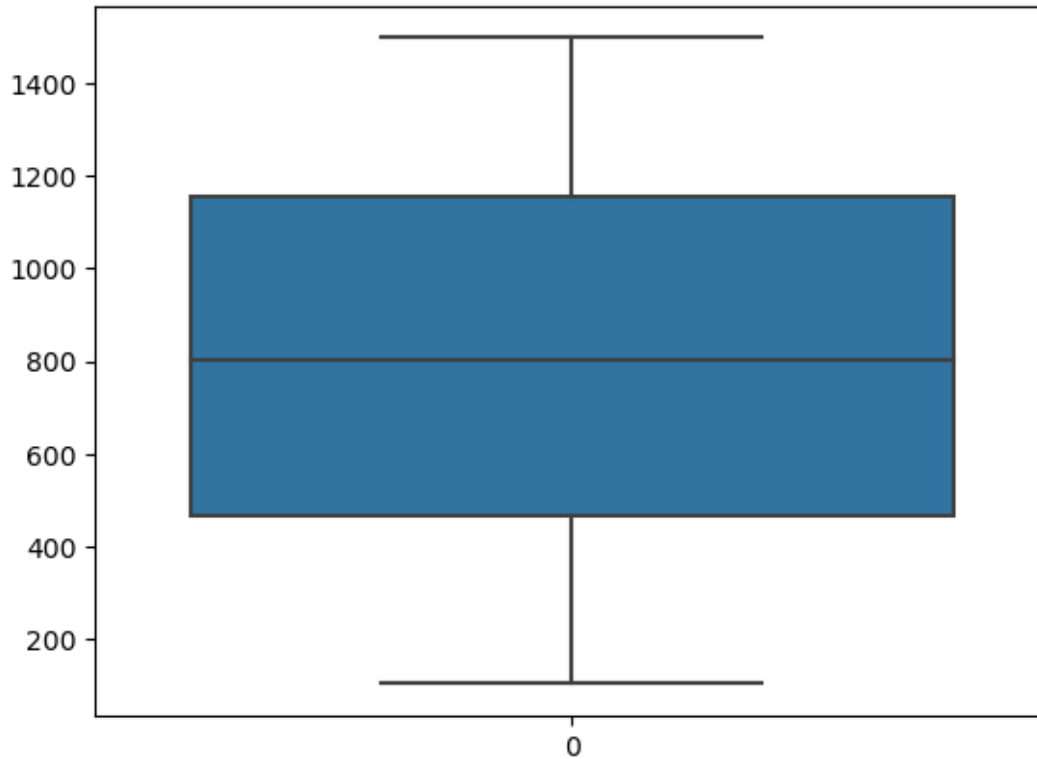
```
[11]: <Axes: >
```





```
[12]: sns.boxplot(data["DailyRate"])
```

```
[12]: <Axes: >
```



```
[13]: data.describe()
```

```
[13]:
```

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

	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	\
count	1470.000000	1470.000000	1470.000000	1470.000000	
mean	1024.865306	2.721769	65.891156	2.729932	
std	602.024335	1.093082	20.329428	0.711561	
min	1.000000	1.000000	30.000000	1.000000	
25%	491.250000	2.000000	48.000000	2.000000	
50%	1020.500000	3.000000	66.000000	3.000000	
75%	1555.750000	4.000000	83.750000	3.000000	
max	2068.000000	4.000000	100.000000	4.000000	

	JobLevel	...	RelationshipSatisfaction	StandardHours	\
count	1470.000000	...	1470.000000	1470.0	
mean	2.063946	...	2.712245	80.0	
std	1.106940	...	1.081209	0.0	
min	1.000000	...	1.000000	80.0	
25%	1.000000	...	2.000000	80.0	
50%	2.000000	...	3.000000	80.0	
75%	3.000000	...	4.000000	80.0	
max	5.000000	...	4.000000	80.0	

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	\
count	1470.000000	1470.000000	1470.000000	
mean	0.793878	11.279592	2.799320	
std	0.852077	7.780782	1.289271	
min	0.000000	0.000000	0.000000	
25%	0.000000	6.000000	2.000000	
50%	1.000000	10.000000	3.000000	
75%	1.000000	15.000000	3.000000	
max	3.000000	40.000000	6.000000	

	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole	\
count	1470.000000	1470.000000	1470.000000	
mean	2.761224	7.008163	4.229252	
std	0.706476	6.126525	3.623137	
min	1.000000	0.000000	0.000000	
25%	2.000000	3.000000	2.000000	
50%	3.000000	5.000000	3.000000	
75%	3.000000	9.000000	7.000000	
max	4.000000	40.000000	18.000000	

	YearsSinceLastPromotion	YearsWithCurrManager
count	1470.000000	1470.000000
mean	2.187755	4.123129
std	3.222430	3.568136
min	0.000000	0.000000
25%	0.000000	2.000000
50%	1.000000	3.000000
75%	3.000000	7.000000
max	15.000000	17.000000

[8 rows x 26 columns]

```
[14]: data.head()
```

```
[14]:   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	EmployeeCount	EmployeeNumber	\
0	1	2	Life Sciences	1	1	
1	8	1	Life Sciences	1	2	
2	2	2	Other	1	4	
3	3	4	Life Sciences	1	5	
4	2	1	Medical	1	7	

	RelationshipSatisfaction	StandardHours	StockOptionLevel	\
0	...	1	80	0
1	...	4	80	1
2	...	2	80	0
3	...	3	80	0
4	...	4	80	1

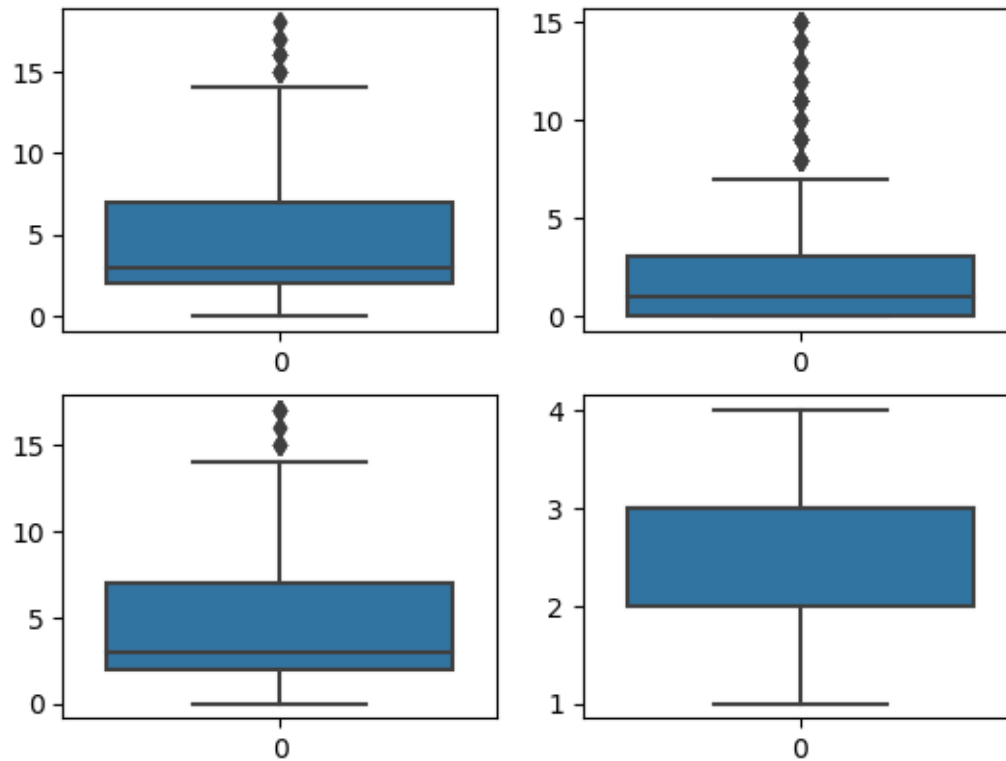
	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	\
0	8	0	1	6	
1	10	3	3	10	
2	7	3	3	0	
3	8	3	3	8	
4	6	3	3	2	

	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 35 columns]

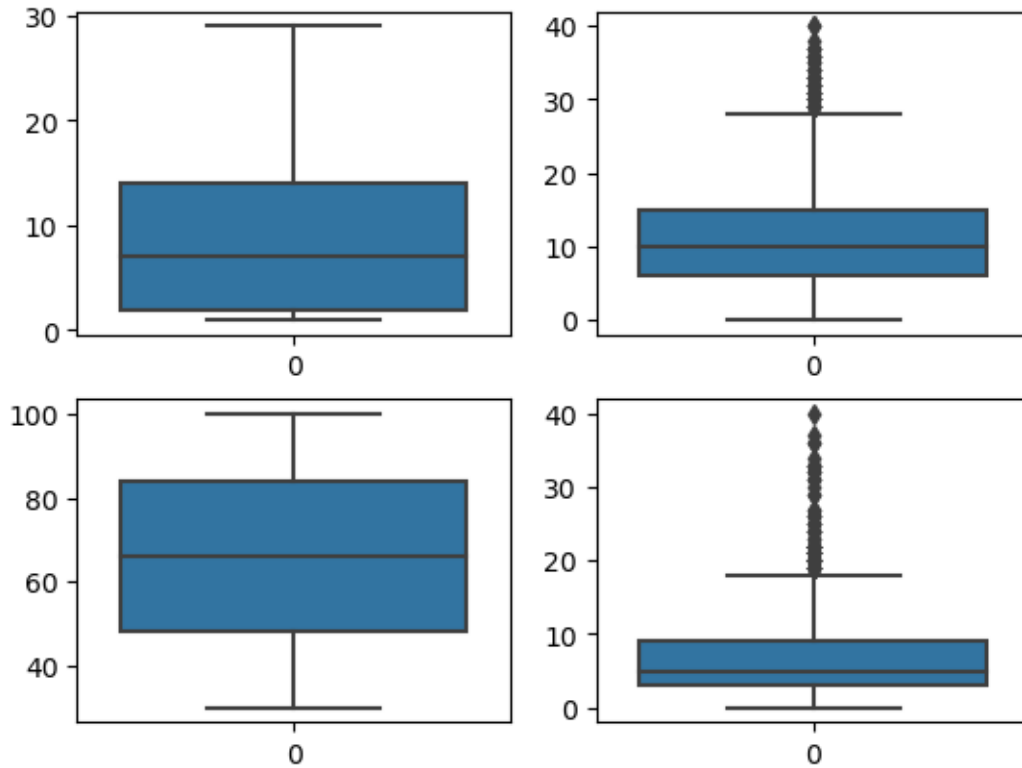
```
[15]: fig, axes = plt.subplots(2,2)
sns.boxplot(data=data["YearsInCurrentRole"],ax=axes[0,0])
sns.boxplot(data=data["YearsSinceLastPromotion"],ax=axes[0,1])
sns.boxplot(data=data["YearsWithCurrManager"],ax=axes[1,0])
sns.boxplot(data=data["WorkLifeBalance"],ax=axes[1,1])
```

[15]: <Axes: >



```
[16]: fig, axes = plt.subplots(2,2)
sns.boxplot(data=data["DistanceFromHome"],ax=axes[0,0])
sns.boxplot(data=data["TotalWorkingYears"],ax=axes[0,1])
sns.boxplot(data=data["HourlyRate"],ax=axes[1,0])
sns.boxplot(data=data["YearsAtCompany"],ax=axes[1,1])
```

[16]: <Axes: >



### 1.3 Handling the Outliers

```
[17]: YearsInCurrentRole_q1 = data.YearsInCurrentRole.quantile(0.25)
YearsInCurrentRole_q3 = data.YearsInCurrentRole.quantile(0.75)
IQR_YearsInCurrentRole=YearsInCurrentRole_q3-YearsInCurrentRole_q1
upperlimit_YearsInCurrentRole=YearsInCurrentRole_q3+1.5*IQR_YearsInCurrentRole
lower_limit_YearsInCurrentRole =YearsInCurrentRole_q1-1.5*IQR_YearsInCurrentRole
median_YearsInCurrentRole=data["YearsInCurrentRole"].median()
data['YearsInCurrentRole'] = np.where(
    (data['YearsInCurrentRole'] > upperlimit_YearsInCurrentRole),
    median_YearsInCurrentRole,
    data['YearsInCurrentRole']
)
```

```
[18]: YearsSinceLastPromotion_q1 = data.YearsSinceLastPromotion.quantile(0.25)
YearsSinceLastPromotion_q3 = data.YearsSinceLastPromotion.quantile(0.75)
IQR_YearsSinceLastPromotion=YearsSinceLastPromotion_q3-YearsSinceLastPromotion_q1
upperlimit_YearsSinceLastPromotion=YearsSinceLastPromotion_q3+1.
↪5*IQR_YearsSinceLastPromotion
lower_limit_YearsSinceLastPromotion =YearsSinceLastPromotion_q1-1.
↪5*IQR_YearsSinceLastPromotion
```

```

median_YearsSinceLastPromotion=data["YearsSinceLastPromotion"].median()
data['YearsSinceLastPromotion'] = np.where(
    (data['YearsSinceLastPromotion'] > upperlimit_YearsSinceLastPromotion),
    median_YearsSinceLastPromotion,
    data['YearsSinceLastPromotion']
)

```

```

[19]: YearsWithCurrManager_q1 = data.YearsWithCurrManager.quantile(0.25)
YearsWithCurrManager_q3 = data.YearsWithCurrManager.quantile(0.75)
IQR_YearsWithCurrManager=YearsWithCurrManager_q3-YearsWithCurrManager_q1
upperlimit_YearsWithCurrManager=YearsWithCurrManager_q3+1.
↳5*IQR_YearsWithCurrManager
lower_limit_YearsWithCurrManager =YearsWithCurrManager_q1-1.
↳5*IQR_YearsWithCurrManager
median_YearsWithCurrManager=data["YearsWithCurrManager"].median()
data['YearsWithCurrManager'] = np.where(
    (data['YearsWithCurrManager'] > upperlimit_YearsWithCurrManager),
    median_YearsWithCurrManager,
    data['YearsWithCurrManager']
)

```

```

[20]: TotalWorkingYears_q1 = data.TotalWorkingYears.quantile(0.25)
TotalWorkingYears_q3 = data.TotalWorkingYears.quantile(0.75)
IQR_TotalWorkingYears=TotalWorkingYears_q3-TotalWorkingYears_q1
upperlimit_TotalWorkingYears=TotalWorkingYears_q3+1.5*IQR_TotalWorkingYears
lower_limit_TotalWorkingYears=TotalWorkingYears_q1-1.5*IQR_TotalWorkingYears
median_TotalWorkingYears=data["TotalWorkingYears"].median()
data['TotalWorkingYears'] = np.where(
    (data['TotalWorkingYears'] > upperlimit_TotalWorkingYears),
    median_TotalWorkingYears,
    data['TotalWorkingYears']
)

```

```

[21]: YearsAtCompany_q1 = data.YearsAtCompany.quantile(0.25)
YearsAtCompany_q3 = data.YearsAtCompany.quantile(0.75)
IQR_YearsAtCompany=YearsAtCompany_q3-YearsAtCompany_q1
upperlimit_YearsAtCompany=YearsAtCompany_q3+1.5*IQR_YearsAtCompany
lower_limit_YearsAtCompany=YearsAtCompany_q1-1.5*IQR_YearsAtCompany
median_YearsAtCompany=data["YearsAtCompany"].median()
data['YearsAtCompany'] = np.where(
    (data['YearsAtCompany'] > upperlimit_YearsAtCompany),
    median_YearsAtCompany,
    data['YearsAtCompany']
)

```

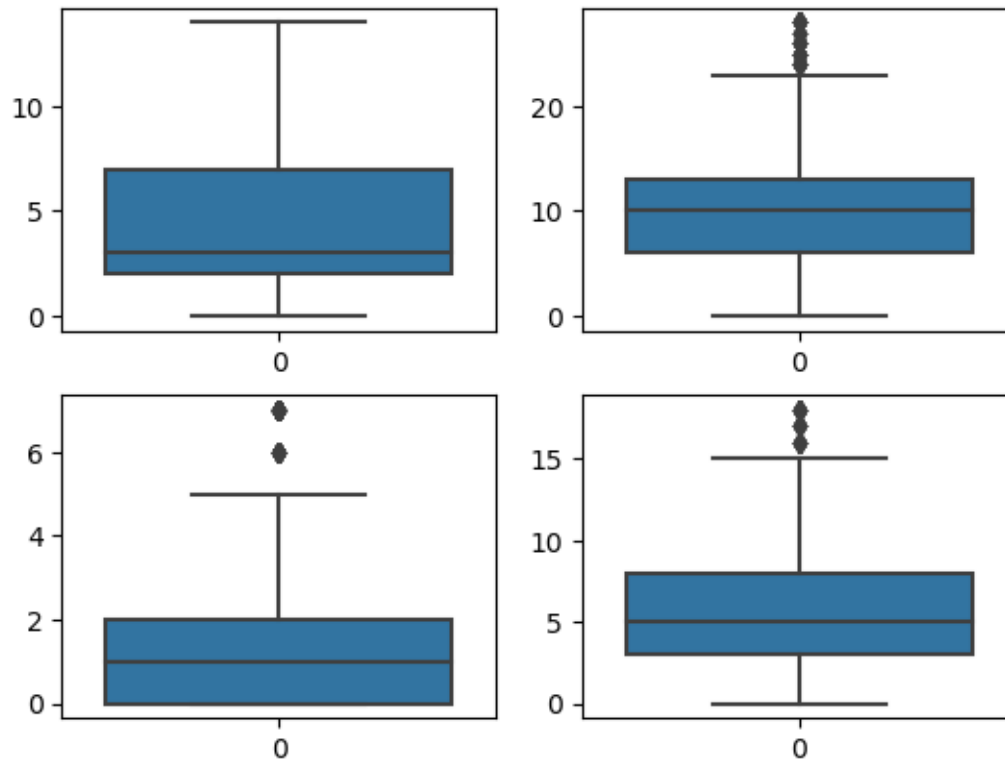
```

[22]: fig, axes = plt.subplots(2,2)
sns.boxplot(data=data["YearsWithCurrManager"],ax=axes[0,0])

```

```
sns.boxplot(data=data["TotalWorkingYears"],ax=axes[0,1])
sns.boxplot(data=data["YearsSinceLastPromotion"],ax=axes[1,0])
sns.boxplot(data=data["YearsAtCompany"],ax=axes[1,1])
```

[22]: <Axes: >



[23]: data.head()

```
[23]:   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   EmployeeCount   EmployeeNumber \
0                1         2 Life Sciences           1           1
1                8         1 Life Sciences           1           2
2                2         2      Other           1           4
3                3         4 Life Sciences           1           5
4                2         1      Medical           1           7
```



	...	RelationshipSatisfaction	StandardHours	StockOptionLevel	\
0	...	1	80	0	
1	...	4	80	1	
2	...	2	80	0	
3	...	3	80	0	
4	...	4	80	1	

	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	\
0	8.0	0	1	6.0	
1	10.0	3	3	10.0	
2	7.0	3	3	0.0	
3	8.0	3	3	8.0	
4	6.0	3	3	2.0	

	YearsInCurrentRole	YearsSinceLastPromotion	YearsWithCurrManager
0	4.0	0.0	5.0
1	7.0	1.0	7.0
2	0.0	0.0	0.0
3	7.0	3.0	0.0
4	2.0	2.0	2.0

[5 rows x 35 columns]

```
[24]: data.drop("EducationField",axis=1,inplace=True)
```

```
[25]: data.head()
```

```
[25]:
```

	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	EmployeeCount	EmployeeNumber	\
0	1	2	1	1	
1	8	1	1	2	
2	2	2	1	4	
3	3	4	1	5	
4	2	1	1	7	

	EnvironmentSatisfaction	...	RelationshipSatisfaction	StandardHours	\
0	2	...	1	80	
1	3	...	4	80	
2	4	...	2	80	
3	4	...	3	80	
4	1	...	4	80	

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	\
0	0	8.0	0	1	
1	1	10.0	3	3	
2	0	7.0	3	3	
3	0	8.0	3	3	
4	1	6.0	3	3	

	YearsAtCompany	YearsInCurrentRole	YearsSinceLastPromotion	\
0	6.0	4.0	0.0	
1	10.0	7.0	1.0	
2	0.0	0.0	0.0	
3	8.0	7.0	3.0	
4	2.0	2.0	2.0	

	YearsWithCurrManager
0	5.0
1	7.0
2	0.0
3	0.0
4	2.0

[5 rows x 34 columns]

```
[26]: data["BusinessTravel"].unique()
```

```
[26]: array(['Travel_Rarely', 'Travel_Frequently', 'Non-Travel'], dtype=object)
```

## 1.4 Splitting the data

```
[27]: y=data["Attrition"]
```

```
[28]: y.head()
```

```
[28]: 0    Yes
      1    No
      2    Yes
      3    No
      4    No
      Name: Attrition, dtype: object
```

```
[29]: data.drop("Attrition",axis=1,inplace=True)
```

```
[30]: data.head()
```

```
[30]:   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	EmployeeCount	EmployeeNumber	\
0	1	2	1	1	
1	8	1	1	2	
2	2	2	1	4	
3	3	4	1	5	
4	2	1	1	7	

	EnvironmentSatisfaction	Gender	...	RelationshipSatisfaction	\
0	2	Female	...	1	
1	3	Male	...	4	
2	4	Male	...	2	
3	4	Female	...	3	
4	1	Male	...	4	

	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	\
0	80	0	8.0	0	
1	80	1	10.0	3	
2	80	0	7.0	3	
3	80	0	8.0	3	
4	80	1	6.0	3	

	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole	\
0	1	6.0	4.0	
1	3	10.0	7.0	
2	3	0.0	0.0	
3	3	8.0	7.0	
4	3	2.0	2.0	

	YearsSinceLastPromotion	YearsWithCurrManager
0	0.0	5.0
1	1.0	7.0
2	0.0	0.0
3	3.0	0.0
4	2.0	2.0

[5 rows x 33 columns]

## 1.5 Encoding

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

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

```
[33]: data["BusinessTravel"]=le.fit_transform(data["BusinessTravel"])
```

```
[34]: data["Department"]=le.fit_transform(data["Department"])
```

```
[35]: data["Gender"]=le.fit_transform(data["Gender"])
```

```
[36]: y=le.fit_transform(y)
```

```
[37]: y
```

```
[37]: array([1, 0, 1, ..., 0, 0, 0])
```

```
[38]: data["JobRole"]=le.fit_transform(data["JobRole"])
```

```
[39]: data["Over18"]=le.fit_transform(data["Over18"])
```

```
[40]: data["MaritalStatus"]=le.fit_transform(data["MaritalStatus"])
```

```
[41]: data["OverTime"]=le.fit_transform(data["OverTime"])
```

```
[42]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 1470 entries, 0 to 1469
```

```
Data columns (total 33 columns):
```

#	Column	Non-Null Count	Dtype
0	Age	1470 non-null	int64
1	BusinessTravel	1470 non-null	int32
2	DailyRate	1470 non-null	int64
3	Department	1470 non-null	int32
4	DistanceFromHome	1470 non-null	int64
5	Education	1470 non-null	int64
6	EmployeeCount	1470 non-null	int64
7	EmployeeNumber	1470 non-null	int64
8	EnvironmentSatisfaction	1470 non-null	int64
9	Gender	1470 non-null	int32
10	HourlyRate	1470 non-null	int64
11	JobInvolvement	1470 non-null	int64
12	JobLevel	1470 non-null	int64
13	JobRole	1470 non-null	int32
14	JobSatisfaction	1470 non-null	int64
15	MaritalStatus	1470 non-null	int32
16	MonthlyIncome	1470 non-null	int64
17	MonthlyRate	1470 non-null	int64
18	NumCompaniesWorked	1470 non-null	int64
19	Over18	1470 non-null	int32
20	OverTime	1470 non-null	int32

21	PercentSalaryHike	1470	non-null	int64
22	PerformanceRating	1470	non-null	int64
23	RelationshipSatisfaction	1470	non-null	int64
24	StandardHours	1470	non-null	int64
25	StockOptionLevel	1470	non-null	int64
26	TotalWorkingYears	1470	non-null	float64
27	TrainingTimesLastYear	1470	non-null	int64
28	WorkLifeBalance	1470	non-null	int64
29	YearsAtCompany	1470	non-null	float64
30	YearsInCurrentRole	1470	non-null	float64
31	YearsSinceLastPromotion	1470	non-null	float64
32	YearsWithCurrManager	1470	non-null	float64

dtypes: float64(5), int32(7), int64(21)  
memory usage: 338.9 KB

## 1.6 Train Test Split

```
[43]: from sklearn.model_selection import train_test_split
      x_train,x_test,y_train,y_test=train_test_split(data,y,test_size=0.
      ↪3,random_state=0)
```

```
[44]: x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
[44]: ((1029, 33), (441, 33), (1029,), (441,))
```

## 1.7 Featuring Scaling

```
[45]: from sklearn.preprocessing import StandardScaler
```

```
[46]: sc=StandardScaler()
```

```
[47]: x_train=sc.fit_transform(x_train)
```

```
[48]: x_test=sc.fit_transform(x_test)
```

# 2 Building the model

## 2.1 Multi Linear Regression

```
[49]: from sklearn.linear_model import LinearRegression
```

```
[50]: lr = LinearRegression()
```

```
[51]: lr.fit(x_train,y_train)
```

```
[51]: LinearRegression()
```

```
[52]: lr.coef_ #slope(m)
```

```
[52]: array([-3.54940447e-02,  7.88352347e-05, -1.70825038e-02,  3.46389690e-02,  
          2.44612841e-02,  3.65668214e-03,  5.37764278e-17, -9.46820520e-03,  
          -4.11203734e-02,  1.06338881e-02, -2.97662154e-03, -3.84864283e-02,  
          -1.52927977e-02, -1.57839139e-02, -3.67252862e-02,  3.35765928e-02,  
          -5.90043558e-03,  5.81099165e-03,  3.78471890e-02,  6.93889390e-18,  
          9.55263279e-02, -2.55800078e-02,  2.01844797e-02, -2.64773510e-02,  
          8.67361738e-19, -1.79286106e-02, -3.30529386e-02, -1.09247807e-02,  
          -3.10631611e-02, -2.47887717e-02, -1.10177742e-02,  2.11897289e-02,  
          -6.60823991e-03])
```

```
[53]: lr.intercept_ #(c)
```

```
[53]: 0.16229348882410102
```

```
[54]: y_pred = lr.predict(x_test)
```

```
[55]: y_pred
```

```
[55]: array([ 1.30302477e-01,  2.17626230e-01,  3.46282415e-01,  5.41382549e-03,  
          4.99292896e-01,  1.01628868e-01,  3.44742777e-01,  1.23994945e-01,  
          -1.60694945e-01,  4.02435622e-01,  1.44159172e-01,  2.67416840e-01,  
          -4.62559536e-02,  5.58671849e-01,  2.81858700e-01,  1.53537792e-02,  
          1.78573363e-01,  2.77532834e-01,  9.37121052e-02,  2.17571624e-01,  
          2.65936178e-01,  1.41499184e-02,  8.36251186e-02,  9.58849826e-02,  
          5.09869963e-01,  2.94764240e-01,  7.85819529e-02,  1.26647773e-01,  
          5.05518902e-01,  8.48456917e-02, -7.97229275e-02,  2.15516993e-02,  
          1.08079105e-01,  3.65998400e-01,  1.24517362e-01,  5.13682786e-02,  
          1.06749689e-01,  6.07640778e-02,  6.66425313e-02,  4.81312859e-02,  
          -1.16761425e-02, -2.97852924e-02,  5.25135582e-02, -1.59076817e-02,  
          -1.71522795e-02,  4.17777714e-01,  3.67341564e-01, -2.14569245e-01,  
          5.47964121e-01,  4.40723777e-01,  1.96701754e-01,  4.42415223e-01,  
          1.45760263e-01,  3.75821843e-01,  4.92762622e-01,  2.95885645e-01,  
          -4.62363391e-02,  3.16337190e-01, -7.90813313e-03,  2.52644685e-01,  
          -3.18239329e-02,  2.83907645e-01,  9.03615010e-02,  1.26934391e-01,  
          3.58670014e-01,  2.40923530e-02,  3.55890111e-01,  1.95961225e-01,  
          1.28554515e-01,  1.18806226e-01, -2.86217094e-02,  3.17635336e-01,  
          1.08017895e-01,  1.25723940e-01,  2.30183307e-01,  9.84315444e-02,  
          9.10911969e-02,  2.72901425e-01,  2.52029723e-01,  4.09210759e-02,  
          -9.10277454e-02, -1.08769544e-02,  1.94114970e-01, -2.25933708e-02,  
          -1.73984898e-02,  1.15587264e-01,  8.36037575e-02,  2.82744685e-03,  
          4.96507732e-02,  2.41862504e-01,  3.14048594e-01,  2.26261102e-01,  
          3.30118359e-01,  2.38527777e-01, -2.16338946e-02,  2.26553579e-01,  
          3.01400098e-01,  2.98806055e-01,  9.89137248e-02,  8.90108718e-02,  
          2.86485256e-01,  5.00403045e-01,  3.03125892e-01, -4.87373316e-03,  
          1.71527163e-01, -5.37529492e-03,  2.54338027e-02,  2.15725447e-01,
```

6.00786752e-02, 1.64813384e-01, 1.09106397e-01, 1.08287462e-01,  
 -3.09499535e-02, 1.96828572e-01, 9.71193504e-02, 3.19061388e-02,  
 1.07934574e-01, 2.33635162e-01, -8.52754375e-02, -7.69198906e-02,  
 2.00624349e-01, 3.35600477e-02, 1.28249663e-01, 6.03012321e-01,  
 5.78155766e-03, -3.07808886e-02, -1.45938525e-01, 2.19398082e-01,  
 2.76229397e-01, 1.67698116e-01, -2.88123044e-03, 2.62341213e-01,  
 4.41290897e-01, 3.95975088e-01, 1.70004873e-01, 4.18305270e-01,  
 4.90462749e-01, 2.02777466e-01, 1.57881421e-01, 3.60759061e-01,  
 2.26021266e-01, 1.45366468e-01, 2.13509469e-01, 2.67909863e-01,  
 3.12986724e-01, -8.02842312e-04, 1.49216491e-01, -1.34599710e-01,  
 2.08537425e-01, 2.79887773e-01, 1.16637429e-01, 2.74165030e-01,  
 5.51651427e-02, 3.41585144e-01, 1.70439326e-01, -7.99466715e-06,  
 -4.10384806e-02, 1.34296605e-01, -1.03707555e-01, -5.60163735e-02,  
 3.36748074e-01, -9.48504896e-02, 2.11704189e-01, 6.18083877e-01,  
 2.03467623e-01, 3.04552682e-01, 1.81990599e-01, 1.84838109e-01,  
 -3.51278477e-03, -8.95239598e-02, 4.14367926e-02, 1.31087001e-01,  
 1.73558095e-01, 1.58265827e-01, -8.67210631e-02, 1.87726385e-01,  
 1.99929237e-01, 1.82109241e-01, 1.03646411e-01, 1.91244072e-01,  
 2.59558194e-03, 1.94666775e-01, -6.08132432e-02, 5.85376580e-01,  
 6.66728668e-02, 4.49620331e-02, 3.30502696e-01, 9.74393000e-02,  
 5.51447175e-01, 1.52212203e-01, 3.58819339e-01, 3.66371593e-01,  
 2.47091987e-01, 5.86970935e-02, 1.28678988e-01, 2.80584025e-01,  
 7.21059443e-02, -8.07006907e-02, 3.39791632e-01, 8.25270203e-02,  
 2.20338157e-01, 2.47703594e-01, 4.97067397e-01, 1.36010592e-01,  
 2.88153807e-01, 4.61306498e-02, 4.52544344e-01, -8.24037634e-02,  
 2.26796295e-01, 1.42129836e-02, 1.62111340e-01, 2.32246950e-01,  
 9.12503556e-02, 1.18866795e-01, 2.12735292e-01, -2.69559828e-02,  
 4.53611463e-02, 1.09618223e-01, 2.64436901e-02, 2.32180310e-01,  
 1.63285101e-01, 2.42669261e-01, 5.44757533e-01, 1.25881866e-01,  
 3.69790740e-01, -8.06922880e-02, 1.41602350e-01, 2.86556696e-01,  
 3.14270745e-01, 1.31598745e-02, -3.56345942e-02, 3.52829749e-01,  
 1.02471015e-01, -2.94842589e-02, 2.57263086e-01, 3.45463501e-01,  
 2.95912601e-01, -1.38481386e-01, 1.39636723e-01, -8.70051183e-03,  
 2.17465256e-01, 2.93583931e-01, 6.93712202e-02, 3.06337299e-01,  
 7.14844408e-02, 2.70630586e-02, 4.05512634e-01, 3.33607408e-01,  
 2.35671783e-03, 2.58266642e-01, 3.28293834e-01, 3.94701029e-01,  
 5.53232045e-01, 8.24575153e-02, 2.60944796e-01, 1.00753288e-01,  
 1.17357357e-02, 5.30783755e-01, 3.34968808e-01, 3.45523339e-01,  
 3.76281748e-01, 5.21936641e-02, 2.44296627e-01, 1.32761428e-01,  
 1.44137632e-01, 1.36122719e-01, -1.50213908e-01, 3.08890370e-01,  
 4.13235863e-01, 1.22281664e-01, 1.68280274e-01, -2.76779470e-02,  
 1.98539478e-01, 8.16150794e-02, -5.55975856e-02, 3.32714819e-02,  
 1.75976835e-01, 3.52080081e-01, 3.23931239e-01, 2.91336298e-01,  
 2.80094379e-01, -7.13513228e-02, 1.84517461e-01, 2.02965882e-01,  
 1.85810057e-01, 2.53361102e-01, -1.32535327e-01, 2.80650525e-01,  
 -1.30523540e-01, 1.68946256e-02, 2.95502669e-01, 6.09961123e-01,  
 1.20850048e-01, 2.99264202e-01, 2.47516030e-01, -1.59794079e-02,

```

3.11830741e-01, 2.55878748e-01, 5.64831824e-01, -8.48534911e-02,
2.27251205e-01, 1.35034035e-01, 1.83392922e-01, 2.67565441e-02,
1.04636218e-01, 1.31700032e-01, 2.07655641e-01, 3.32733438e-01,
-2.01238580e-01, -9.51544438e-02, 4.94665237e-02, -7.32543548e-03,
2.09656376e-01, 2.75173006e-01, 2.27685889e-01, 2.80495892e-01,
2.36503496e-01, 7.47322815e-02, 1.20343079e-01, -1.10102575e-02,
2.66505865e-01, 1.34380235e-01, 3.83985037e-01, 3.27038149e-01,
1.92718914e-01, 1.38040637e-01, 3.13471437e-01, 3.03369774e-01,
-1.79198125e-01, 1.06960652e-01, -3.23223785e-03, 3.45401694e-01,
1.36157048e-01, 3.82744300e-01, 2.46142802e-01, -1.70015194e-01,
2.55048994e-01, -2.28290648e-01, 3.36992530e-01, 1.49968922e-01,
1.17049406e-01, 4.83890032e-01, 1.46067187e-01, 4.18288393e-01,
1.16743641e-01, 9.56873703e-03, 2.34800404e-01, 1.36313424e-01,
7.74987760e-03, 2.34345722e-01, 6.10723706e-02, 3.41757613e-01,
1.59047657e-01, 9.67999468e-02, 2.32540380e-01, -8.19460772e-02,
1.93362458e-01, 1.91174979e-01, -6.58901852e-03, 2.65125211e-01,
-2.67132109e-01, -3.75025206e-02, 1.82935564e-01, 1.80163736e-02,
1.97392018e-01, 2.06959344e-01, -1.09857091e-01, 3.54059406e-01,
7.26868239e-02, 2.36653676e-01, -8.20686416e-02, 3.20816139e-01,
1.60355866e-02, 2.39463166e-01, 3.15828470e-01, 1.23580438e-01,
8.23103381e-03, 9.37109487e-02, 4.76095644e-01, -5.67913774e-02,
3.21054350e-01, 1.87697945e-02, 1.84780396e-01, 2.15401165e-01,
-2.02255818e-01, 2.52065992e-01, -1.51719947e-01, 3.50282752e-01,
-9.84668529e-02, 2.14753151e-01, 2.46151971e-02, 2.09633480e-01,
9.48374386e-02, 3.12075361e-01, 4.06959059e-01, -2.26761393e-02,
7.65002089e-02, 3.06988986e-01, 4.37484499e-02, 1.09186169e-01,
4.21237022e-01, 1.49225126e-01, 4.45704191e-02, 3.37793947e-02,
9.97831984e-02, -5.30137196e-02, 3.47224412e-01, 1.03944090e-01,
-4.09963236e-02, -5.78838569e-02, 3.09525724e-01, 1.02060488e-01,
2.09447907e-01, 4.10525192e-01, 3.33473319e-01, 1.80971041e-01,
2.76792072e-01, 2.86132531e-01, 2.62476320e-01, -1.83021903e-02,
2.36094900e-01, 1.54018489e-01, 6.36220924e-02, 6.18224799e-03,
1.85057193e-02, 7.69476922e-02, 1.34623859e-01, 1.87169316e-01,
2.36666289e-01, -1.82114662e-01, 2.98547908e-01, 1.73398527e-01,
-8.87118635e-02, 3.51838607e-02, 1.35598577e-01, 1.70085191e-01,
1.69932034e-01, 2.29056852e-01, 2.15573570e-01, 1.04403736e-01,
-8.21467550e-02])

```

```
[56]: y_test
```

```

[56]: array([0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 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, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 1, 0, 1, 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, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,

```



```

0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 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,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 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, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0])

```

## 2.2 Logistic Regression

```
[57]: from sklearn.linear_model import LogisticRegression
```

```
[58]: lg=LogisticRegression()
```

```
[59]: lg.fit(x_train,y_train)
```

```
[59]: LogisticRegression()
```

```
[60]: y_pred_lg=lg.predict(x_test)
```

```
[61]: y_pred
```

```
[61]: array([ 1.30302477e-01,  2.17626230e-01,  3.46282415e-01,  5.41382549e-03,
  4.99292896e-01,  1.01628868e-01,  3.44742777e-01,  1.23994945e-01,
 -1.60694945e-01,  4.02435622e-01,  1.44159172e-01,  2.67416840e-01,
 -4.62559536e-02,  5.58671849e-01,  2.81858700e-01,  1.53537792e-02,
  1.78573363e-01,  2.77532834e-01,  9.37121052e-02,  2.17571624e-01,
  2.65936178e-01,  1.41499184e-02,  8.36251186e-02,  9.58849826e-02,
  5.09869963e-01,  2.94764240e-01,  7.85819529e-02,  1.26647773e-01,
  5.05518902e-01,  8.48456917e-02, -7.97229275e-02,  2.15516993e-02,
  1.08079105e-01,  3.65998400e-01,  1.24517362e-01,  5.13682786e-02,
  1.06749689e-01,  6.07640778e-02,  6.66425313e-02,  4.81312859e-02,
 -1.16761425e-02, -2.97852924e-02,  5.25135582e-02, -1.59076817e-02,
 -1.71522795e-02,  4.17777714e-01,  3.67341564e-01, -2.14569245e-01,
  5.47964121e-01,  4.40723777e-01,  1.96701754e-01,  4.42415223e-01,
  1.45760263e-01,  3.75821843e-01,  4.92762622e-01,  2.95885645e-01,
 -4.62363391e-02,  3.16337190e-01, -7.90813313e-03,  2.52644685e-01,
 -3.18239329e-02,  2.83907645e-01,  9.03615010e-02,  1.26934391e-01,
  3.58670014e-01,  2.40923530e-02,  3.55890111e-01,  1.95961225e-01,
  1.28554515e-01,  1.18806226e-01, -2.86217094e-02,  3.17635336e-01,

```

1.08017895e-01, 1.25723940e-01, 2.30183307e-01, 9.84315444e-02,  
 9.10911969e-02, 2.72901425e-01, 2.52029723e-01, 4.09210759e-02,  
 -9.10277454e-02, -1.08769544e-02, 1.94114970e-01, -2.25933708e-02,  
 -1.73984898e-02, 1.15587264e-01, 8.36037575e-02, 2.82744685e-03,  
 4.96507732e-02, 2.41862504e-01, 3.14048594e-01, 2.26261102e-01,  
 3.30118359e-01, 2.38527777e-01, -2.16338946e-02, 2.26553579e-01,  
 3.01400098e-01, 2.98806055e-01, 9.89137248e-02, 8.90108718e-02,  
 2.86485256e-01, 5.00403045e-01, 3.03125892e-01, -4.87373316e-03,  
 1.71527163e-01, -5.37529492e-03, 2.54338027e-02, 2.15725447e-01,  
 6.00786752e-02, 1.64813384e-01, 1.09106397e-01, 1.08287462e-01,  
 -3.09499535e-02, 1.96828572e-01, 9.71193504e-02, 3.19061388e-02,  
 1.07934574e-01, 2.33635162e-01, -8.52754375e-02, -7.69198906e-02,  
 2.00624349e-01, 3.35600477e-02, 1.28249663e-01, 6.03012321e-01,  
 5.78155766e-03, -3.07808886e-02, -1.45938525e-01, 2.19398082e-01,  
 2.76229397e-01, 1.67698116e-01, -2.88123044e-03, 2.62341213e-01,  
 4.41290897e-01, 3.95975088e-01, 1.70004873e-01, 4.18305270e-01,  
 4.90462749e-01, 2.02777466e-01, 1.57881421e-01, 3.60759061e-01,  
 2.26021266e-01, 1.45366468e-01, 2.13509469e-01, 2.67909863e-01,  
 3.12986724e-01, -8.02842312e-04, 1.49216491e-01, -1.34599710e-01,  
 2.08537425e-01, 2.79887773e-01, 1.16637429e-01, 2.74165030e-01,  
 5.51651427e-02, 3.41585144e-01, 1.70439326e-01, -7.99466715e-06,  
 -4.10384806e-02, 1.34296605e-01, -1.03707555e-01, -5.60163735e-02,  
 3.36748074e-01, -9.48504896e-02, 2.11704189e-01, 6.18083877e-01,  
 2.03467623e-01, 3.04552682e-01, 1.81990599e-01, 1.84838109e-01,  
 -3.51278477e-03, -8.95239598e-02, 4.14367926e-02, 1.31087001e-01,  
 1.73558095e-01, 1.58265827e-01, -8.67210631e-02, 1.87726385e-01,  
 1.99929237e-01, 1.82109241e-01, 1.03646411e-01, 1.91244072e-01,  
 2.59558194e-03, 1.94666775e-01, -6.08132432e-02, 5.85376580e-01,  
 6.66728668e-02, 4.49620331e-02, 3.30502696e-01, 9.74393000e-02,  
 5.51447175e-01, 1.52212203e-01, 3.58819339e-01, 3.66371593e-01,  
 2.47091987e-01, 5.86970935e-02, 1.28678988e-01, 2.80584025e-01,  
 7.21059443e-02, -8.07006907e-02, 3.39791632e-01, 8.25270203e-02,  
 2.20338157e-01, 2.47703594e-01, 4.97067397e-01, 1.36010592e-01,  
 2.88153807e-01, 4.61306498e-02, 4.52544344e-01, -8.24037634e-02,  
 2.26796295e-01, 1.42129836e-02, 1.62111340e-01, 2.32246950e-01,  
 9.12503556e-02, 1.18866795e-01, 2.12735292e-01, -2.69559828e-02,  
 4.53611463e-02, 1.09618223e-01, 2.64436901e-02, 2.32180310e-01,  
 1.63285101e-01, 2.42669261e-01, 5.44757533e-01, 1.25881866e-01,  
 3.69790740e-01, -8.06922880e-02, 1.41602350e-01, 2.86556696e-01,  
 3.14270745e-01, 1.31598745e-02, -3.56345942e-02, 3.52829749e-01,  
 1.02471015e-01, -2.94842589e-02, 2.57263086e-01, 3.45463501e-01,  
 2.95912601e-01, -1.38481386e-01, 1.39636723e-01, -8.70051183e-03,  
 2.17465256e-01, 2.93583931e-01, 6.93712202e-02, 3.06337299e-01,  
 7.14844408e-02, 2.70630586e-02, 4.05512634e-01, 3.33607408e-01,  
 2.35671783e-03, 2.58266642e-01, 3.28293834e-01, 3.94701029e-01,  
 5.53232045e-01, 8.24575153e-02, 2.60944796e-01, 1.00753288e-01,  
 1.17357357e-02, 5.30783755e-01, 3.34968808e-01, 3.45523339e-01,

3.76281748e-01, 5.21936641e-02, 2.44296627e-01, 1.32761428e-01,  
 1.44137632e-01, 1.36122719e-01, -1.50213908e-01, 3.08890370e-01,  
 4.13235863e-01, 1.22281664e-01, 1.68280274e-01, -2.76779470e-02,  
 1.98539478e-01, 8.16150794e-02, -5.55975856e-02, 3.32714819e-02,  
 1.75976835e-01, 3.52080081e-01, 3.23931239e-01, 2.91336298e-01,  
 2.80094379e-01, -7.13513228e-02, 1.84517461e-01, 2.02965882e-01,  
 1.85810057e-01, 2.53361102e-01, -1.32535327e-01, 2.80650525e-01,  
 -1.30523540e-01, 1.68946256e-02, 2.95502669e-01, 6.09961123e-01,  
 1.20850048e-01, 2.99264202e-01, 2.47516030e-01, -1.59794079e-02,  
 3.11830741e-01, 2.55878748e-01, 5.64831824e-01, -8.48534911e-02,  
 2.27251205e-01, 1.35034035e-01, 1.83392922e-01, 2.67565441e-02,  
 1.04636218e-01, 1.31700032e-01, 2.07655641e-01, 3.32733438e-01,  
 -2.01238580e-01, -9.51544438e-02, 4.94665237e-02, -7.32543548e-03,  
 2.09656376e-01, 2.75173006e-01, 2.27685889e-01, 2.80495892e-01,  
 2.36503496e-01, 7.47322815e-02, 1.20343079e-01, -1.10102575e-02,  
 2.66505865e-01, 1.34380235e-01, 3.83985037e-01, 3.27038149e-01,  
 1.92718914e-01, 1.38040637e-01, 3.13471437e-01, 3.03369774e-01,  
 -1.79198125e-01, 1.06960652e-01, -3.23223785e-03, 3.45401694e-01,  
 1.36157048e-01, 3.82744300e-01, 2.46142802e-01, -1.70015194e-01,  
 2.55048994e-01, -2.28290648e-01, 3.36992530e-01, 1.49968922e-01,  
 1.17049406e-01, 4.83890032e-01, 1.46067187e-01, 4.18288393e-01,  
 1.16743641e-01, 9.56873703e-03, 2.34800404e-01, 1.36313424e-01,  
 7.74987760e-03, 2.34345722e-01, 6.10723706e-02, 3.41757613e-01,  
 1.59047657e-01, 9.67999468e-02, 2.32540380e-01, -8.19460772e-02,  
 1.93362458e-01, 1.91174979e-01, -6.58901852e-03, 2.65125211e-01,  
 -2.67132109e-01, -3.75025206e-02, 1.82935564e-01, 1.80163736e-02,  
 1.97392018e-01, 2.06959344e-01, -1.09857091e-01, 3.54059406e-01,  
 7.26868239e-02, 2.36653676e-01, -8.20686416e-02, 3.20816139e-01,  
 1.60355866e-02, 2.39463166e-01, 3.15828470e-01, 1.23580438e-01,  
 8.23103381e-03, 9.37109487e-02, 4.76095644e-01, -5.67913774e-02,  
 3.21054350e-01, 1.87697945e-02, 1.84780396e-01, 2.15401165e-01,  
 -2.02255818e-01, 2.52065992e-01, -1.51719947e-01, 3.50282752e-01,  
 -9.84668529e-02, 2.14753151e-01, 2.46151971e-02, 2.09633480e-01,  
 9.48374386e-02, 3.12075361e-01, 4.06959059e-01, -2.26761393e-02,  
 7.65002089e-02, 3.06988986e-01, 4.37484499e-02, 1.09186169e-01,  
 4.21237022e-01, 1.49225126e-01, 4.45704191e-02, 3.37793947e-02,  
 9.97831984e-02, -5.30137196e-02, 3.47224412e-01, 1.03944090e-01,  
 -4.09963236e-02, -5.78838569e-02, 3.09525724e-01, 1.02060488e-01,  
 2.09447907e-01, 4.10525192e-01, 3.33473319e-01, 1.80971041e-01,  
 2.76792072e-01, 2.86132531e-01, 2.62476320e-01, -1.83021903e-02,  
 2.36094900e-01, 1.54018489e-01, 6.36220924e-02, 6.18224799e-03,  
 1.85057193e-02, 7.69476922e-02, 1.34623859e-01, 1.87169316e-01,  
 2.36666289e-01, -1.82114662e-01, 2.98547908e-01, 1.73398527e-01,  
 -8.87118635e-02, 3.51838607e-02, 1.35598577e-01, 1.70085191e-01,  
 1.69932034e-01, 2.29056852e-01, 2.15573570e-01, 1.04403736e-01,  
 -8.21467550e-02])

```
[62]: y_test
```

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

```
[63]: score = lg.score(x_test, y_test)
      print(score)
```

```
0.8820861678004536
```

## 2.3 Confusion Matrix

```
[64]: from sklearn import metrics
      cm = metrics.confusion_matrix(y_test, y_pred_lg)
      print(cm)
```

```
[[366   5]
 [ 47  23]]
```

## 2.4 Ridge and Lasso

```
[65]: from sklearn.linear_model import Ridge
      from sklearn.model_selection import GridSearchCV
```

```
[66]: rg=Ridge()
```

```
[67]: parametres={"alpha": [1,2,3,5,10,20,30,40,60,70,80,90]}
      ridgecv=GridSearchCV(rg,parametres,scoring="neg_mean_squared_error",cv=5)
      ridgecv.fit(x_train,y_train)
```

```
[67]: GridSearchCV(cv=5, estimator=Ridge(),  
                param_grid={'alpha': [1, 2, 3, 5, 10, 20, 30, 40, 60, 70, 80, 90]},  
                scoring='neg_mean_squared_error')
```

```
[68]: print(ridgecv.best_params_)
```

```
{'alpha': 90}
```

```
[69]: print(ridgecv.best_score_)
```

```
-0.11390621139234185
```

```
[70]: y_pred_rg=ridgecv.predict(x_test)
```

```
[71]: y_pred_rg
```

```
[71]: array([ 1.34413485e-01,  2.22561818e-01,  3.41692977e-01,  3.88209867e-03,  
            4.84617338e-01,  1.16361483e-01,  3.30449743e-01,  1.27358807e-01,  
           -1.34442619e-01,  3.77692888e-01,  1.33001445e-01,  2.69898751e-01,  
           -2.54707392e-02,  5.25771894e-01,  2.67543514e-01,  2.78725024e-02,  
            1.82233111e-01,  2.78896415e-01,  9.12689699e-02,  2.11494641e-01,  
            2.70103341e-01,  8.44922044e-03,  8.74746722e-02,  1.05348798e-01,  
            4.87749940e-01,  2.83080512e-01,  8.80556209e-02,  1.23817268e-01,  
            4.82185624e-01,  9.34824523e-02, -7.16448509e-02,  4.07003104e-02,  
            1.08437994e-01,  3.42151399e-01,  1.22270929e-01,  6.85889862e-02,  
            1.06690533e-01,  7.08689637e-02,  7.51570276e-02,  6.05829413e-02,  
            1.08782897e-02, -6.91368661e-03,  5.83191600e-02, -1.54680056e-02,  
           -4.02267475e-03,  4.08010612e-01,  3.43668700e-01, -1.83519405e-01,  
            5.29536511e-01,  4.27646098e-01,  1.95234877e-01,  4.25012930e-01,  
            1.40754410e-01,  3.52173952e-01,  4.70372694e-01,  2.89240343e-01,  
           -3.11642726e-02,  3.04206456e-01,  9.89337674e-03,  2.44569884e-01,  
           -1.40249115e-02,  2.75133912e-01,  8.64669565e-02,  1.24214885e-01,  
            3.48994545e-01,  3.41026778e-02,  3.40548051e-01,  1.95847356e-01,  
            1.30040885e-01,  1.32259137e-01, -2.34680143e-02,  3.04595468e-01,  
            1.12452197e-01,  1.30525275e-01,  2.19329505e-01,  9.44722098e-02,  
            9.98185782e-02,  2.60042486e-01,  2.51475715e-01,  4.59039018e-02,  
           -7.94007856e-02, -7.05812314e-03,  2.04344419e-01, -3.97180151e-03,  
           -5.91286905e-03,  1.26797761e-01,  8.02495203e-02,  2.55422079e-02,  
            4.65384158e-02,  2.32985240e-01,  3.16063931e-01,  2.02833301e-01,  
            3.14235904e-01,  2.33427101e-01, -1.42446708e-02,  2.24789285e-01,  
            2.94719863e-01,  2.94698323e-01,  1.19907108e-01,  9.47152062e-02,  
            2.86026178e-01,  4.75925979e-01,  2.87802013e-01,  6.72561468e-03,  
            1.65013565e-01,  1.72887026e-02,  3.34684186e-02,  2.15466121e-01,  
            7.50317322e-02,  1.67646673e-01,  1.16585544e-01,  1.07157808e-01,  
           -1.84689359e-02,  1.86217544e-01,  1.16586463e-01,  4.67201201e-02,  
            1.11060472e-01,  2.27053971e-01, -7.00247692e-02, -5.81070776e-02,  
            2.03141688e-01,  4.69029664e-02,  1.31525768e-01,  5.66738022e-01,  
            2.41883060e-02, -3.41250985e-02, -1.13904557e-01,  2.18572744e-01,
```

2.60568042e-01, 1.65533667e-01, -5.94078459e-05, 2.60009384e-01,  
 4.20709666e-01, 3.71031267e-01, 1.70250288e-01, 4.03052216e-01,  
 4.67312765e-01, 1.98845366e-01, 1.55005619e-01, 3.41505080e-01,  
 2.20024496e-01, 1.40989758e-01, 1.97796963e-01, 2.57841889e-01,  
 2.99122317e-01, 9.24907038e-03, 1.39162817e-01, -1.13916709e-01,  
 1.97670909e-01, 2.70864780e-01, 1.22454317e-01, 2.58893294e-01,  
 6.78818374e-02, 3.08485027e-01, 1.49347982e-01, 2.01436659e-02,  
 -3.30262214e-02, 1.44305312e-01, -8.99199978e-02, -3.74712872e-02,  
 3.10198738e-01, -7.96862570e-02, 2.18579680e-01, 5.85363859e-01,  
 1.98166099e-01, 3.02558934e-01, 1.82182301e-01, 1.84955080e-01,  
 1.83694574e-02, -7.41419216e-02, 4.48013268e-02, 1.38405390e-01,  
 1.84013774e-01, 1.60373463e-01, -6.83819091e-02, 2.00146771e-01,  
 1.97563797e-01, 1.73505024e-01, 1.01481984e-01, 1.83169586e-01,  
 1.99747065e-02, 1.81881922e-01, -5.23948254e-02, 5.46171171e-01,  
 6.66114639e-02, 5.88865384e-02, 3.17247692e-01, 9.77721299e-02,  
 5.25297461e-01, 1.62566350e-01, 3.51341492e-01, 3.58324715e-01,  
 2.37059552e-01, 8.05788438e-02, 1.36041888e-01, 2.66653277e-01,  
 7.95513973e-02, -6.96788172e-02, 3.29442074e-01, 8.93231393e-02,  
 2.16673846e-01, 2.50725892e-01, 4.72995721e-01, 1.26285837e-01,  
 2.72059331e-01, 6.13056795e-02, 4.38912502e-01, -7.79381284e-02,  
 2.09974643e-01, 2.20746796e-02, 1.56186553e-01, 2.26485767e-01,  
 9.61150570e-02, 1.27870464e-01, 2.13995902e-01, -9.95070059e-03,  
 2.59908614e-02, 1.24499158e-01, 3.31256404e-02, 2.39369272e-01,  
 1.48870840e-01, 2.49438253e-01, 5.25239856e-01, 1.25104891e-01,  
 3.65711314e-01, -5.96554519e-02, 1.45443911e-01, 2.80327834e-01,  
 3.15149040e-01, 2.25038913e-02, -2.55968584e-02, 3.39893959e-01,  
 1.13068264e-01, -1.40778596e-02, 2.41548652e-01, 3.30468114e-01,  
 2.79352111e-01, -1.11653133e-01, 1.30812123e-01, 5.94136334e-03,  
 2.15791900e-01, 2.89769560e-01, 6.35971718e-02, 2.98116673e-01,  
 5.90198982e-02, 4.13352130e-02, 3.77978672e-01, 3.28916149e-01,  
 3.93836061e-03, 2.47850806e-01, 3.23692453e-01, 3.69738410e-01,  
 5.25445397e-01, 8.81102278e-02, 2.60937425e-01, 9.86284476e-02,  
 1.33675781e-02, 5.03508194e-01, 3.13852304e-01, 3.23698856e-01,  
 3.56168397e-01, 6.40677131e-02, 2.39875950e-01, 1.30293453e-01,  
 1.49885837e-01, 1.46778013e-01, -1.27871947e-01, 3.00062088e-01,  
 3.83140060e-01, 1.33561747e-01, 1.76445188e-01, -2.02405714e-02,  
 2.05300799e-01, 8.08917915e-02, -2.89389327e-02, 3.88975789e-02,  
 1.66754600e-01, 3.46231726e-01, 3.01325806e-01, 2.82141156e-01,  
 2.72386022e-01, -4.84039332e-02, 1.90411095e-01, 1.89810174e-01,  
 1.67705712e-01, 2.58007480e-01, -1.08871386e-01, 2.70056893e-01,  
 -1.13365420e-01, 3.41754165e-02, 2.92189798e-01, 5.71351976e-01,  
 1.25404974e-01, 2.97241061e-01, 2.29580043e-01, 2.16006118e-03,  
 2.94123343e-01, 2.48418344e-01, 5.44089373e-01, -6.68570736e-02,  
 2.24222741e-01, 1.35207932e-01, 1.91240682e-01, 3.87158365e-02,  
 1.17416027e-01, 1.20780070e-01, 2.19845528e-01, 3.11363894e-01,  
 -1.70611843e-01, -6.89093044e-02, 5.87182022e-02, 1.46846582e-02,  
 2.07924051e-01, 2.57017951e-01, 2.33929422e-01, 2.81772193e-01,

```

2.28214280e-01, 8.56533063e-02, 1.28782763e-01, 6.89964545e-03,
2.66888647e-01, 1.25060071e-01, 3.71449815e-01, 3.23080605e-01,
1.80691831e-01, 1.35736688e-01, 3.05937337e-01, 3.02505887e-01,
-1.61842349e-01, 1.22990350e-01, 1.33351902e-02, 3.42072173e-01,
1.35302335e-01, 3.71921241e-01, 2.43987619e-01, -1.42777272e-01,
2.54178774e-01, -1.97769860e-01, 3.22194537e-01, 1.46753524e-01,
1.28207748e-01, 4.59813547e-01, 1.49345212e-01, 3.97978765e-01,
1.11492990e-01, 2.94090069e-02, 2.30676290e-01, 1.19192017e-01,
1.49461455e-02, 2.42284371e-01, 7.22361156e-02, 3.33852369e-01,
1.61213354e-01, 9.69685794e-02, 2.32264965e-01, -6.93181380e-02,
1.86467739e-01, 2.03098589e-01, -1.10349710e-02, 2.63095846e-01,
-2.48406147e-01, -3.25418955e-02, 1.74487006e-01, 2.62780720e-02,
1.91428194e-01, 2.03493779e-01, -8.84696022e-02, 3.35631012e-01,
6.29476544e-02, 2.28818932e-01, -7.72255471e-02, 3.05353195e-01,
3.63634109e-02, 2.30128273e-01, 3.03522210e-01, 1.05913376e-01,
1.26693452e-02, 9.53511494e-02, 4.52766233e-01, -4.37470263e-02,
3.05687630e-01, 3.57706117e-02, 1.82867743e-01, 2.10106289e-01,
-1.71378996e-01, 2.60157245e-01, -1.38655420e-01, 3.36603939e-01,
-7.65297319e-02, 2.15165094e-01, 3.72947326e-02, 1.96608549e-01,
1.07172893e-01, 3.07687901e-01, 3.97760529e-01, 1.06797074e-03,
8.12866229e-02, 2.95445495e-01, 5.47994817e-02, 1.13818287e-01,
4.07117263e-01, 1.48860323e-01, 3.88471838e-02, 3.79029267e-02,
1.09895981e-01, -4.30946471e-02, 3.30298512e-01, 1.07254284e-01,
-1.13032643e-02, -3.69192632e-02, 2.87732288e-01, 9.91961213e-02,
2.12225886e-01, 3.88660531e-01, 3.15623317e-01, 1.80996998e-01,
2.69970366e-01, 2.81850174e-01, 2.49972461e-01, -2.33065542e-03,
2.34240860e-01, 1.51536128e-01, 6.56810225e-02, 1.35221573e-02,
3.03956323e-02, 9.22075626e-02, 1.28297232e-01, 2.04669352e-01,
2.26917512e-01, -1.62627965e-01, 2.95984225e-01, 1.80934145e-01,
-6.34810776e-02, 4.36092057e-02, 1.39814157e-01, 1.72029014e-01,
1.65538329e-01, 2.24411690e-01, 2.15315070e-01, 1.16342630e-01,
-6.24745967e-02])

```

```
[72]: y_test
```

```

[72]: array([0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
1, 1, 0, 1, 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, 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, 1,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 1, 0, 0, 0, 0, 0, 0, 0, 0,

```

```

0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 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, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0])

```

```

[73]: from sklearn import metrics
      print(metrics.r2_score(y_test,y_pred_rg))
      print(metrics.r2_score(y_train,ridgecv.predict(x_train)))

```

```

0.21073458438815884
0.2061567210285108

```

## 2.5 Lasso

```

[74]: from sklearn.linear_model import Lasso
      from sklearn.model_selection import GridSearchCV

```

```

[75]: la=Ridge()

```

```

[76]: parametres={"alpha": [1,2,3,5,10,20,30,40,60,70,80,90]}
      ridgecv=GridSearchCV(la,parametres,scoring="neg_mean_squared_error",cv=5)
      ridgecv.fit(x_train,y_train)

```

```

[76]: GridSearchCV(cv=5, estimator=Ridge(),
                  param_grid={'alpha': [1, 2, 3, 5, 10, 20, 30, 40, 60, 70, 80, 90]},
                  scoring='neg_mean_squared_error')

```

```

[77]: print(ridgecv.best_params_)

```

```

{'alpha': 90}

```

```

[78]: print(ridgecv.best_score_)

```

```

-0.11390621139234185

```

```

[79]: y_pred_la=ridgecv.predict(x_test)

```

```

[80]: y_pred_la

```

```

[80]: array([ 1.34413485e-01,  2.22561818e-01,  3.41692977e-01,  3.88209867e-03,
            4.84617338e-01,  1.16361483e-01,  3.30449743e-01,  1.27358807e-01,
           -1.34442619e-01,  3.77692888e-01,  1.33001445e-01,  2.69898751e-01,
           -2.54707392e-02,  5.25771894e-01,  2.67543514e-01,  2.78725024e-02,
            1.82233111e-01,  2.78896415e-01,  9.12689699e-02,  2.11494641e-01,

```



2.70103341e-01, 8.44922044e-03, 8.74746722e-02, 1.05348798e-01,  
 4.87749940e-01, 2.83080512e-01, 8.80556209e-02, 1.23817268e-01,  
 4.82185624e-01, 9.34824523e-02, -7.16448509e-02, 4.07003104e-02,  
 1.08437994e-01, 3.42151399e-01, 1.22270929e-01, 6.85889862e-02,  
 1.06690533e-01, 7.08689637e-02, 7.51570276e-02, 6.05829413e-02,  
 1.08782897e-02, -6.91368661e-03, 5.83191600e-02, -1.54680056e-02,  
 -4.02267475e-03, 4.08010612e-01, 3.43668700e-01, -1.83519405e-01,  
 5.29536511e-01, 4.27646098e-01, 1.95234877e-01, 4.25012930e-01,  
 1.40754410e-01, 3.52173952e-01, 4.70372694e-01, 2.89240343e-01,  
 -3.11642726e-02, 3.04206456e-01, 9.89337674e-03, 2.44569884e-01,  
 -1.40249115e-02, 2.75133912e-01, 8.64669565e-02, 1.24214885e-01,  
 3.48994545e-01, 3.41026778e-02, 3.40548051e-01, 1.95847356e-01,  
 1.30040885e-01, 1.32259137e-01, -2.34680143e-02, 3.04595468e-01,  
 1.12452197e-01, 1.30525275e-01, 2.19329505e-01, 9.44722098e-02,  
 9.98185782e-02, 2.60042486e-01, 2.51475715e-01, 4.59039018e-02,  
 -7.94007856e-02, -7.05812314e-03, 2.04344419e-01, -3.97180151e-03,  
 -5.91286905e-03, 1.26797761e-01, 8.02495203e-02, 2.55422079e-02,  
 4.65384158e-02, 2.32985240e-01, 3.16063931e-01, 2.02833301e-01,  
 3.14235904e-01, 2.33427101e-01, -1.42446708e-02, 2.24789285e-01,  
 2.94719863e-01, 2.94698323e-01, 1.19907108e-01, 9.47152062e-02,  
 2.86026178e-01, 4.75925979e-01, 2.87802013e-01, 6.72561468e-03,  
 1.65013565e-01, 1.72887026e-02, 3.34684186e-02, 2.15466121e-01,  
 7.50317322e-02, 1.67646673e-01, 1.16585544e-01, 1.07157808e-01,  
 -1.84689359e-02, 1.86217544e-01, 1.16586463e-01, 4.67201201e-02,  
 1.11060472e-01, 2.27053971e-01, -7.00247692e-02, -5.81070776e-02,  
 2.03141688e-01, 4.69029664e-02, 1.31525768e-01, 5.66738022e-01,  
 2.41883060e-02, -3.41250985e-02, -1.13904557e-01, 2.18572744e-01,  
 2.60568042e-01, 1.65533667e-01, -5.94078459e-05, 2.60009384e-01,  
 4.20709666e-01, 3.71031267e-01, 1.70250288e-01, 4.03052216e-01,  
 4.67312765e-01, 1.98845366e-01, 1.55005619e-01, 3.41505080e-01,  
 2.20024496e-01, 1.40989758e-01, 1.97796963e-01, 2.57841889e-01,  
 2.99122317e-01, 9.24907038e-03, 1.39162817e-01, -1.13916709e-01,  
 1.97670909e-01, 2.70864780e-01, 1.22454317e-01, 2.58893294e-01,  
 6.78818374e-02, 3.08485027e-01, 1.49347982e-01, 2.01436659e-02,  
 -3.30262214e-02, 1.44305312e-01, -8.99199978e-02, -3.74712872e-02,  
 3.10198738e-01, -7.96862570e-02, 2.18579680e-01, 5.85363859e-01,  
 1.98166099e-01, 3.02558934e-01, 1.82182301e-01, 1.84955080e-01,  
 1.83694574e-02, -7.41419216e-02, 4.48013268e-02, 1.38405390e-01,  
 1.84013774e-01, 1.60373463e-01, -6.83819091e-02, 2.00146771e-01,  
 1.97563797e-01, 1.73505024e-01, 1.01481984e-01, 1.83169586e-01,  
 1.99747065e-02, 1.81881922e-01, -5.23948254e-02, 5.46171171e-01,  
 6.66114639e-02, 5.88865384e-02, 3.17247692e-01, 9.77721299e-02,  
 5.25297461e-01, 1.62566350e-01, 3.51341492e-01, 3.58324715e-01,  
 2.37059552e-01, 8.05788438e-02, 1.36041888e-01, 2.66653277e-01,  
 7.95513973e-02, -6.96788172e-02, 3.29442074e-01, 8.93231393e-02,  
 2.16673846e-01, 2.50725892e-01, 4.72995721e-01, 1.26285837e-01,  
 2.72059331e-01, 6.13056795e-02, 4.38912502e-01, -7.79381284e-02,

2.09974643e-01, 2.20746796e-02, 1.56186553e-01, 2.26485767e-01,  
 9.61150570e-02, 1.27870464e-01, 2.13995902e-01, -9.95070059e-03,  
 2.59908614e-02, 1.24499158e-01, 3.31256404e-02, 2.39369272e-01,  
 1.48870840e-01, 2.49438253e-01, 5.25239856e-01, 1.25104891e-01,  
 3.65711314e-01, -5.96554519e-02, 1.45443911e-01, 2.80327834e-01,  
 3.15149040e-01, 2.25038913e-02, -2.55968584e-02, 3.39893959e-01,  
 1.13068264e-01, -1.40778596e-02, 2.41548652e-01, 3.30468114e-01,  
 2.79352111e-01, -1.11653133e-01, 1.30812123e-01, 5.94136334e-03,  
 2.15791900e-01, 2.89769560e-01, 6.35971718e-02, 2.98116673e-01,  
 5.90198982e-02, 4.13352130e-02, 3.77978672e-01, 3.28916149e-01,  
 3.93836061e-03, 2.47850806e-01, 3.23692453e-01, 3.69738410e-01,  
 5.25445397e-01, 8.81102278e-02, 2.60937425e-01, 9.86284476e-02,  
 1.33675781e-02, 5.03508194e-01, 3.13852304e-01, 3.23698856e-01,  
 3.56168397e-01, 6.40677131e-02, 2.39875950e-01, 1.30293453e-01,  
 1.49885837e-01, 1.46778013e-01, -1.27871947e-01, 3.00062088e-01,  
 3.83140060e-01, 1.33561747e-01, 1.76445188e-01, -2.02405714e-02,  
 2.05300799e-01, 8.08917915e-02, -2.89389327e-02, 3.88975789e-02,  
 1.66754600e-01, 3.46231726e-01, 3.01325806e-01, 2.82141156e-01,  
 2.72386022e-01, -4.84039332e-02, 1.90411095e-01, 1.89810174e-01,  
 1.67705712e-01, 2.58007480e-01, -1.08871386e-01, 2.70056893e-01,  
 -1.13365420e-01, 3.41754165e-02, 2.92189798e-01, 5.71351976e-01,  
 1.25404974e-01, 2.97241061e-01, 2.29580043e-01, 2.16006118e-03,  
 2.94123343e-01, 2.48418344e-01, 5.44089373e-01, -6.68570736e-02,  
 2.24222741e-01, 1.35207932e-01, 1.91240682e-01, 3.87158365e-02,  
 1.17416027e-01, 1.20780070e-01, 2.19845528e-01, 3.11363894e-01,  
 -1.70611843e-01, -6.89093044e-02, 5.87182022e-02, 1.46846582e-02,  
 2.07924051e-01, 2.57017951e-01, 2.33929422e-01, 2.81772193e-01,  
 2.28214280e-01, 8.56533063e-02, 1.28782763e-01, 6.89964545e-03,  
 2.66888647e-01, 1.25060071e-01, 3.71449815e-01, 3.23080605e-01,  
 1.80691831e-01, 1.35736688e-01, 3.05937337e-01, 3.02505887e-01,  
 -1.61842349e-01, 1.22990350e-01, 1.33351902e-02, 3.42072173e-01,  
 1.35302335e-01, 3.71921241e-01, 2.43987619e-01, -1.42777272e-01,  
 2.54178774e-01, -1.97769860e-01, 3.22194537e-01, 1.46753524e-01,  
 1.28207748e-01, 4.59813547e-01, 1.49345212e-01, 3.97978765e-01,  
 1.11492990e-01, 2.94090069e-02, 2.30676290e-01, 1.19192017e-01,  
 1.49461455e-02, 2.42284371e-01, 7.22361156e-02, 3.33852369e-01,  
 1.61213354e-01, 9.69685794e-02, 2.32264965e-01, -6.93181380e-02,  
 1.86467739e-01, 2.03098589e-01, -1.10349710e-02, 2.63095846e-01,  
 -2.48406147e-01, -3.25418955e-02, 1.74487006e-01, 2.62780720e-02,  
 1.91428194e-01, 2.03493779e-01, -8.84696022e-02, 3.35631012e-01,  
 6.29476544e-02, 2.28818932e-01, -7.72255471e-02, 3.05353195e-01,  
 3.63634109e-02, 2.30128273e-01, 3.03522210e-01, 1.05913376e-01,  
 1.26693452e-02, 9.53511494e-02, 4.52766233e-01, -4.37470263e-02,  
 3.05687630e-01, 3.57706117e-02, 1.82867743e-01, 2.10106289e-01,  
 -1.71378996e-01, 2.60157245e-01, -1.38655420e-01, 3.36603939e-01,  
 -7.65297319e-02, 2.15165094e-01, 3.72947326e-02, 1.96608549e-01,  
 1.07172893e-01, 3.07687901e-01, 3.97760529e-01, 1.06797074e-03,

```

8.12866229e-02, 2.95445495e-01, 5.47994817e-02, 1.13818287e-01,
4.07117263e-01, 1.48860323e-01, 3.88471838e-02, 3.79029267e-02,
1.09895981e-01, -4.30946471e-02, 3.30298512e-01, 1.07254284e-01,
-1.13032643e-02, -3.69192632e-02, 2.87732288e-01, 9.91961213e-02,
2.12225886e-01, 3.88660531e-01, 3.15623317e-01, 1.80996998e-01,
2.69970366e-01, 2.81850174e-01, 2.49972461e-01, -2.33065542e-03,
2.34240860e-01, 1.51536128e-01, 6.56810225e-02, 1.35221573e-02,
3.03956323e-02, 9.22075626e-02, 1.28297232e-01, 2.04669352e-01,
2.26917512e-01, -1.62627965e-01, 2.95984225e-01, 1.80934145e-01,
-6.34810776e-02, 4.36092057e-02, 1.39814157e-01, 1.72029014e-01,
1.65538329e-01, 2.24411690e-01, 2.15315070e-01, 1.16342630e-01,
-6.24745967e-02])

```

```

[81]: from sklearn import metrics
      print(metrics.r2_score(y_test,y_pred_la))
      print(metrics.r2_score(y_train,ridgecv.predict(x_train)))

```

```

0.21073458438815884
0.2061567210285108

```

## 2.6 Decision Tree

```

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

```

```

[83]: dtc.fit(x_train,y_train)

```

```

[83]: DecisionTreeClassifier()

```

```

[84]: pred=dtc.predict(x_test)

```

```

[85]: pred

```

```

[85]: array([0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
          0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
          0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
          0, 1, 1, 0, 0, 0, 0, 1, 1, 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, 1, 0, 1, 0, 0, 1, 0,
          0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1,
          0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 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, 1, 0, 0,

```

```

0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 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, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0])

```

```
[86]: y_test
```

```

[86]: array([0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 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, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
1, 1, 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, 0, 0, 1, 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, 1,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 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, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1,
0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0])

```

```

[87]: #Accuracy score
from sklearn.metrics import
    accuracy_score, confusion_matrix, classification_report, roc_auc_score, roc_curve

```

```
[88]: accuracy_score(y_test, pred)
```

```
[88]: 0.7755102040816326
```

```
[89]: confusion_matrix(y_test, pred)
```

```

[89]: array([[320,  51],
        [ 48,  22]], dtype=int64)

```

```
[90]: pd.crosstab(y_test, pred)
```

```
[90]: col_0    0    1
      row_0
      0    320  51
      1     48  22
```

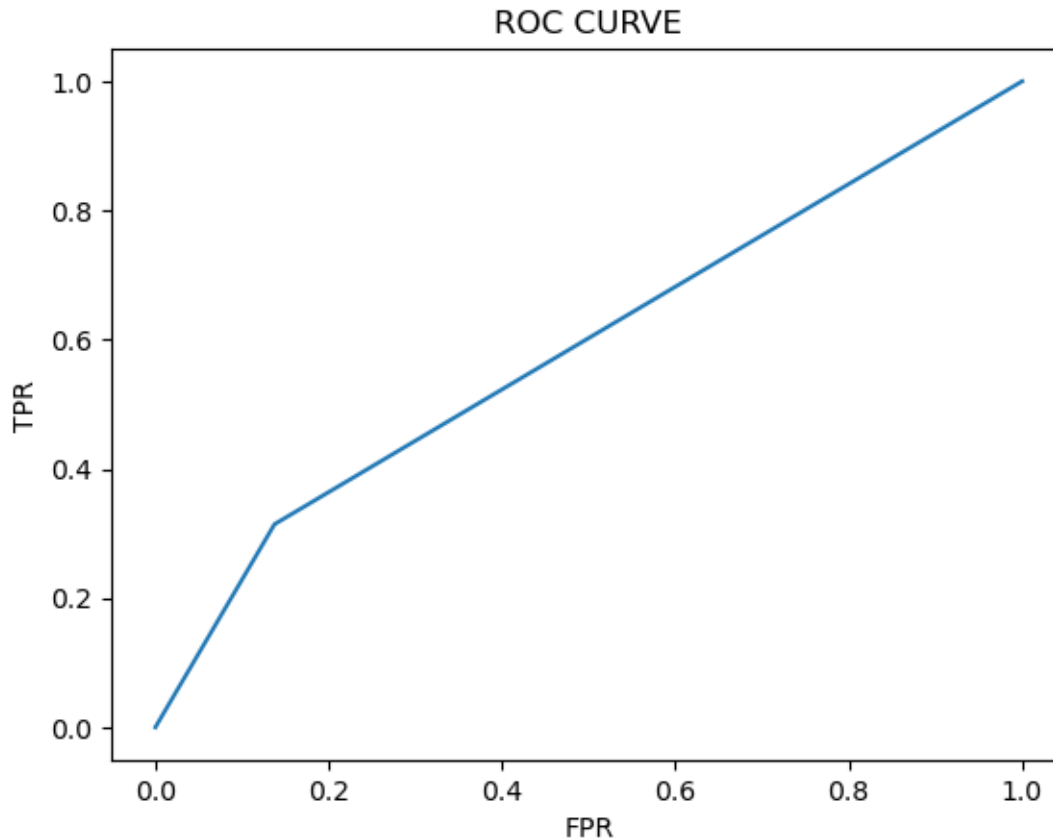
```
[91]: print(classification_report(y_test,pred))
```

	precision	recall	f1-score	support
0	0.87	0.86	0.87	371
1	0.30	0.31	0.31	70
accuracy			0.78	441
macro avg	0.59	0.59	0.59	441
weighted avg	0.78	0.78	0.78	441

```
[92]: probability=dtc.predict_proba(x_test)[: ,1]
```

```
[93]: # roc_curve
      fpr,tpr,threshholds = roc_curve(y_test,probability)
```

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



## 2.7 Random Forest

```
[95]: from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

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

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

```
[98]: rfc_cv= GridSearchCV(rfc,param_grid=forest_params,cv=10,scoring="accuracy")
```

```
[99]: rfc_cv.fit(x_train,y_train)
```

C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model\_selection\\_validation.py:425: FitFailedWarning:  
50 fits failed out of a total of 700.  
The score on these train-test partitions for these parameters will be set to nan.  
If these failures are not expected, you can try to debug them by setting error\_score='raise'.

Below are more details about the failures:

-----  
50 fits failed with the following error:

Traceback (most recent call last):

```
File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model_selection\_validation.py", line 732, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py", line 1144, in wrapper
    estimator._validate_params()
File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py", line 637, in _validate_params
    validate_parameter_constraints(
File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\utils\_param_validation.py", line 95, in validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'max_features' parameter of RandomForestClassifier must be an int in the range [1, inf), a float in the range (0.0, 1.0], a str among {'sqrt', 'log2'} or None. Got 0 instead.
```

```
warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:976: UserWarning: One or more of the
test scores are non-finite: [          nan 0.84353703 0.84645917 0.85229393
0.85226537 0.85517799
0.85517799 0.85612983 0.84545022 0.85517799 0.85033314 0.85518751
0.8541976  0.85227489          nan 0.8445079  0.84937179 0.847411
0.85324576 0.85032362 0.85322673 0.84936227 0.85227489 0.85227489
0.85614887 0.85031411 0.84740148 0.85227489          nan 0.84256615
0.84546926 0.85422616 0.84935275 0.84644013 0.85712926 0.85227489
0.85615839 0.85422616 0.85614887 0.85227489 0.85131354 0.84838188
          nan 0.84256615 0.85032362 0.85422616 0.84935275 0.85033314
0.85325528 0.85032362 0.84644013 0.85225585 0.85227489 0.85420712
0.85517799 0.85031411          nan 0.84645917 0.84936227 0.85422616
0.85225585 0.85130402 0.85130402 0.85418808 0.85128498 0.85323625
0.85224634 0.84935275 0.85420712 0.85711974]
warnings.warn(
```

```
[99]: GridSearchCV(cv=10, estimator=RandomForestClassifier(),
                  param_grid=[{'max_depth': [10, 11, 12, 13, 14],
                                'max_features': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
                                                  12, 13]}],
                  scoring='accuracy')
```

```
[100]: pred=rfc_cv.predict(x_test)
```

```
[101]: print(classification_report(y_test,pred))
```

	precision	recall	f1-score	support
0	0.87	0.99	0.92	371
1	0.74	0.20	0.31	70
accuracy			0.86	441
macro avg	0.80	0.59	0.62	441
weighted avg	0.85	0.86	0.83	441

```
[102]: rfc_cv.best_params_
```

```
[102]: {'max_depth': 12, 'max_features': 6}
```

```
[103]: rfc_cv.best_score_
```

```
[103]: 0.8571292594707784
```