In [7]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import math
from sklearn.linear_model import LogisticRegression
```

In [8]:
```
heart =pd.read_csv("C:\\Users\\HP\\Desktop\\LINEAR PROGRAMMING CENTRALS\\heart
```

In [13]:
```
heart
```

Out[13]:

	rest_bp	chest_pain	thalassemia	age	fasting_bs	max_hr	exercise_angina	gender	st_slop
0	106	3	0	67	0	142	0	0	
1	120	2	0	50	0	158	0	0	
2	126	3	2	35	0	156	1	1	
3	150	3	2	63	0	154	0	0	
4	140	3	2	46	0	120	1	1	
...	...	...	...	...	...	...	...	...	.
94	150	3	2	60	0	157	0	0	
95	170	3	1	58	1	146	1	0	
96	130	1	0	45	0	175	0	0	
97	130	3	0	61	0	169	0	0	
98	150	3	2	58	0	111	1	1	

99 rows × 14 columns

In [15]:
```
X=np.array(heart[["age","chest_pain"]])
Y=np.array(heart["diagnosis"])
```

In [16]:
```
X
```

Out[16]:
```
array([[67,  3],
       [50,  2],
       [35,  3],
       [63,  3],
       [46,  3],
       [57,  3],
       [49,  1],
       [52,  2],
       [59,  3],
       [64,  0],
       [44,  3],
       [52,  2],
       [47,  3],
       [53,  2],
       [57,  3],
       [67,  3],
       [62,  1],
       [66,  3],
       [52,  1],
       [65,  2],
       [43,  3],
       [56,  0],
       [68,  2],
       [42,  0],
       [46,  2],
       [54,  1],
       [44,  1],
       [65,  2],
       [67,  2],
       [59,  3],
       [48,  2],
       [45,  1],
       [54,  3],
       [58,  2],
       [48,  3],
       [43,  3],
       [64,  3],
       [57,  2],
       [56,  2],
       [43,  3],
       [58,  2],
       [60,  3],
       [57,  3],
       [66,  2],
       [51,  2],
       [60,  3],
       [64,  3],
       [67,  2],
       [70,  3],
       [51,  2],
       [68,  2],
       [54,  2],
       [51,  0],
       [41,  1],
```

```
       [63,  1],
       [56,  1],
       [45,  3],
       [51,  2],
       [62,  3],
       [54,  2],
       [39,  2],
       [49,  1],
       [61,  3],
       [51,  3],
       [57,  3],
       [55,  3],
       [70,  2],
       [65,  3],
       [35,  3],
       [59,  3],
       [67,  3],
       [60,  3],
       [39,  2],
       [46,  3],
       [54,  3],
       [47,  3],
       [56,  3],
       [60,  3],
       [68,  2],
       [54,  3],
       [51,  2],
       [63,  3],
       [43,  3],
       [55,  3],
       [57,  3],
       [44,  2],
       [49,  2],
       [58,  2],
       [39,  2],
       [42,  2],
       [66,  3],
       [63,  2],
       [46,  1],
       [59,  0],
       [60,  3],
       [58,  3],
       [45,  1],
       [61,  3],
       [58,  3]], dtype=int64)
```

In [17]:
```
Y
```

Out[17]:
```
array([0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
       0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1], dtype=int64)
```

In [18]:
```
# splitting the data
X_train, X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random_stat
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(74, 2)
(25, 2)
(74,)
(25,)
```

In [19]:
```
heart.isnull()
```

Out[19]:

	rest_bp	chest_pain	thalassemia	age	fasting_bs	max_hr	exercise_angina	gender	st_slo
0	False	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	False	Fa
...	...	...	...	...	...	...	...	...	
94	False	False	False	False	False	False	False	False	Fa
95	False	False	False	False	False	False	False	False	Fa
96	False	False	False	False	False	False	False	False	Fa
97	False	False	False	False	False	False	False	False	Fa
98	False	False	False	False	False	False	False	False	Fa

99 rows × 14 columns

In [20]:
```
heart.isnull().sum()
```

Out[20]:
```
rest_bp            0
chest_pain         0
thalassemia        0
age                0
fasting_bs         0
max_hr             0
exercise_angina    0
gender             0
st_slope           0
cholesterol        0
st_depression      0
rest_ecg           0
num_vessels        0
diagnosis          0
dtype: int64
```

In [21]:
```
heart.head(5)
```

Out[21]:

	rest_bp	chest_pain	thalassemia	age	fasting_bs	max_hr	exercise_angina	gender	st_slope
0	106	3	0	67	0	142	0	0	0
1	120	2	0	50	0	158	0	0	1
2	126	3	2	35	0	156	1	1	0
3	150	3	2	63	0	154	0	0	1
4	140	3	2	46	0	120	1	1	1

In [22]:
```
X_train, X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.25,random_stat
logmodel=LogisticRegression()
logmodel.fit(X_train,Y_train)
```

Out[22]:
```
▾ LogisticRegression
LogisticRegression()
```

In [24]:
```
model.fit(X_train,Y_train)
```

Out[24]:
```
▾ LogisticRegression
LogisticRegression()
```

In [25]:
```
Y_pred=model.predict(X_test)
Y_pred
```

Out[25]:
```
array([0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0,
       0, 0, 0], dtype=int64)
```

In [26]:
```
from sklearn.metrics import precision_score,recall_score,accuracy_score,f1_sco
```

In [28]:
```
accuracy_score(Y_test,Y_pred)
```

Out[28]:
```
0.76
```

In [29]:
```
precision_score(Y_test,Y_pred)
```

Out[29]:
```
0.8
```

In [30]:
```
recall_score(Y_test,Y_pred)
```

Out[30]:
```
0.6666666666666666
```

In [31]:
```
f1_score(Y_test,Y_pred)
```

Out[31]:
```
0.7272727272727272
```

In [32]:
```
Edward =LogisticRegression()
Edward
```

Out[32]:
```
▾ LogisticRegression

LogisticRegression()
```

In [34]:
```
from sklearn.model_selection import GridSearchCV
param_grid={
    'penalty':['11','12','elasticnet',None],
    'solver':['lbfgs','liblinear','newton-cg','sag','saga'],
    'C':[1],
    'dual':[True,False]
}
param_grid
```

Out[34]:
```
{'penalty': ['11', '12', 'elasticnet', None],
 'solver': ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga'],
 'C': [1],
 'dual': [True, False]}
```

In [35]:
```
grid_search=GridSearchCV(Edward,param_grid,cv=5)
grid_search
```

Out[35]:
```
▸          GridSearchCV

▸ estimator: LogisticRegression

      ▸ LogisticRegression
```

In [ ]:

In [ ]: