In [167]:

```
%cd C:\Users\manoj\Downloads
```

```
C:\Users\manoj\Downloads
```

In [168]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
import scipy.stats
```

In [169]:

```
df=pd.read_csv('diabetes.csv')
```

In [170]:

```
X=np.array(df.iloc[:,:8])
```

In [171]:

```
y=np.array(df.iloc[:,8])
```

In [172]:

```
print(X)
```

```
[[  6.    148.     72.    ...  33.6     0.627  50.   ]
 [  1.     85.     66.    ...  26.6     0.351  31.   ]
 [  8.    183.     64.    ...  23.3     0.672  32.   ]
 ...
 [  5.    121.     72.    ...  26.2     0.245  30.   ]
 [  1.    126.     60.    ...  30.1     0.349  47.   ]
 [  1.     93.     70.    ...  30.4     0.315  23.   ]]
```

In [173]:

```
print(y)
```

```
[1 0 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0 0
 1 1 1 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 1 0
 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1
 1 0 0 1 1 1 0 0 0 1 0 0 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 1 0 1 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 0
 1 1 1 1 1 0 0 1 1 0 1 0 1 1 1 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 1 1 0 1 1 1 1
 0 0 0 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0
 1 0 1 0 0 1 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 1 1 1 0 0
 1 0 1 0 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1 0 0 0 1
 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 1 1 0 0 1 0 0 1 0 0 1
 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0 1 0 1 1 0 1 0 1 0 1
 0 1 1 0 0 0 0 1 1 0 1 0 1 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1
 1 1 0 0 1 0 0 1 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 1
 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0
 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0 1 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1 0 1 0 1 0 1 0
 1 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0
 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 1 1 1 0
 1 1 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 1 0 1 0 1 0 1 1 0 0 0 0 1 1
 0 0 0 1 0 1 1 0 0 1 0 0 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1
 1 0 0 1 0 0 1 0 1 1 1 0 0 1 1 1 0 1 0 1 0 1 0 0 0 0 1 0]
```

In [174]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=0)
```

In [175]:

```
from sklearn.linear_model import LogisticRegression
```

In [176]:

```
logreg = LogisticRegression(max_iter=300)
```

In [177]:

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

Out[177]:

```
LogisticRegression(max_iter=300)
```

In [178]:

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

In [179]:

```
print(y_pred)
```

```
[1 0 0 1 0 0 1 1 0 0 1 1 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 1 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 1 1 1 1 0 0 0 0 0 0 1
 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0
 0 1 0 1 1 0 1 0 1 0 0 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 0
 0 0 0 1 0 0 1 0 1 0 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0
 0 1 0 0 0 0 0]
```

In [180]:

```
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

Out[180]:

```
array([[117,  13],
       [ 26,  36]], dtype=int64)
```

In [181]:

```
df['Outcome'].value_counts()
```

Out[181]:

```
0    500
1    268
Name: Outcome, dtype: int64
```

In [182]:

```
count_0=0
count_1=0
for i in y_test:
    if i==0:
        count_0=count_0+1
    elif i==1:
            count_1=count_1+1
print('ze',count_0,'on',count_1)
```

```
ze 130 on 62
```

In [183]:

```
print(metrics.classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.82      0.90      0.86       130
           1       0.73      0.58      0.65        62

    accuracy                           0.80       192
   macro avg       0.78      0.74      0.75       192
weighted avg       0.79      0.80      0.79       192
```

In [184]:

```
metrics.accuracy_score(y_test, y_pred)
```

Out[184]:

```
0.796875
```

In [185]:

```
print(logreg.predict_proba(X_test))
```

```
 [0.35172145 0.64827855]
 [0.23591566 0.76408434]
 [0.98020668 0.01979332]
 [0.75919349 0.24080651]
 [0.15429253 0.84570747]
 [0.71884417 0.28115583]
 [0.8131489  0.1868511 ]
 [0.89175116 0.10824884]
 [0.76722454 0.23277546]
 [0.89305364 0.10694636]
 [0.89679272 0.10320728]

 [0.78433041 0.21566959]
 [0.82656811 0.17343189]
 [0.74976188 0.25023812]
 [0.45238413 0.54761587]
 [0.87421574 0.12578426]
 [0.61170116 0.38829884]
 [0.88653043 0.11346957]
 [0.88352202 0.11647798]
```

In [186]:

```
print(y_pred)
```

```
[1 0 0 1 0 0 1 1 0 0 1 1 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 1 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 1 1 1 1 0 0 0 0 0 0 1
 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0
 0 1 0 1 1 0 1 0 1 0 0 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 0
 0 0 0 1 0 0 1 0 1 0 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0
 0 1 0 0 0 0 0]
```

In [187]:

```
#for haberman
%cd C:\Users\manoj\Downloads
```

```
C:\Users\manoj\Downloads
```

In [188]:

```
df=pd.read_csv('haberman.csv')
```

In [189]:

```
df=pd.read_csv('haberman.csv',names=['Agr','Yearofoperation','Positivenodes','Survivalrate'
df
```

Out[189]:

	Agr	Yearofoperation	Positivenodes	Survivalrate
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1
...	...	...	...	...
301	75	62	1	1
302	76	67	0	1
303	77	65	3	1
304	78	65	1	2
305	83	58	2	2

306 rows × 4 columns

In [190]:

```
X=np.array(df.iloc[:,:3])
```

In [191]:

```
y=np.array(df.iloc[:,3])
```

In [192]:

```
print(X)
```

```
 [65 64  0]
 [65 67  1]
 [66 58  0]
 [66 61 13]
 [66 58  0]
 [66 58  1]
 [66 68  0]
 [67 64  8]
 [67 63  1]
 [67 66  0]
 [67 66  0]
 [67 61  0]
 [67 65  0]
 [68 67  0]
 [68 68  0]
 [69 67  8]
 [69 60  0]
 [69 65  0]
 [69 66  0]
```

In [193]:

```
print(y)
```

```
[1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 2 1 1
 1 1 1 1 1 1 2 2 2 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 2
 2 2 1 1 1 1 2 2 2 1 1 1 1 1 1 2 2 2 2 1 1 1 2 2 2 1 1 1 1 1 1 1 2 2 2 1
 1 1 1 2 2 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 2 2 2 1 1
 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 1 1 1
 1 1 1 1 1 2 2 1 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1
 1 2 2 1 1 1 1 2 2 2 1 1 1 1 1 1 2 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2
 2 2 2 1 1 1 1 1 1 2 2 1 1 1 2 2 1 1 1 1 1 1 2 1 1 1 2 2 1 1 1 1 1 1 2 1 1
 1 1 1 2 1 1 1 1 2 2]
```

In [194]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=0)
```

In [195]:

```
from sklearn.linear_model import LogisticRegression
```

In [196]:

```
aim = LogisticRegression()
```

In [197]:

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

Out[197]:

```
LogisticRegression()
```

In [198]:

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

In [199]:

```
print(y_pred)
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2
 1 1 1]
```

In [200]:

```
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

Out[200]:

```
array([[45,  1],
       [29,  2]], dtype=int64)
```

In [201]:

```
df['Survivalrate'].value_counts()
```

Out[201]:

```
1    225
2     81
Name: Survivalrate, dtype: int64
```

In [202]:

```
count_1=0
count_2=0
for i in y_test:
    if i==1:
        count_1=count_1+1
    elif i==2:
            count_2=count_2+1
print('ones',count_1,'twos',count_2)
```

```
ones 46 twos 31
```

In [203]:

```
print(metrics.classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.61      0.98      0.75        46
           2       0.67      0.06      0.12        31

    accuracy                           0.61        77
   macro avg       0.64      0.52      0.43        77
weighted avg       0.63      0.61      0.50        77
```

In [204]:

```
metrics.accuracy_score(y_test, y_pred)
```

Out[204]:

```
0.6103896103896104
```

In [205]:

```
print(aim.predict_proba(X_test))
```

```
[[0.78443735 0.21556265]
 [0.65970536 0.34029464]
 [0.80116172 0.19883828]
 [0.81941146 0.18058854]
 [0.70141742 0.29858258]
 [0.66705534 0.33294466]
 [0.83147882 0.16852118]
 [0.50377239 0.49622761]
 [0.53138826 0.46861174]
 [0.81554251 0.18445749]
 [0.77194963 0.22805037]
 [0.85864188 0.14135812]
 [0.80669521 0.19330479]
 [0.80538218 0.19461782]
 [0.55300694 0.44699306]
 [0.81910055 0.18089945]
 [0.5230679  0.4769321 ]
 [0.84625553 0.15374447]
 [0.85399135 0.14600865]
 [0.90414258 0.09585742]
 [0.89420344 0.10579656]
 [0.80700811 0.19299189]
 [0.81910055 0.18089945]
 [0.83562662 0.16437338]
 [0.68572991 0.31427009]
 [0.84764655 0.15235345]
 [0.82676971 0.17323029]
 [0.8428514  0.1571486 ]
 [0.85760219 0.14239781]
 [0.84442923 0.15557077]
 [0.84757621 0.15242379]
 [0.82140295 0.17859705]
 [0.74549095 0.25450905]
 [0.7896929  0.2103071 ]
 [0.88496601 0.11503399]
 [0.82759231 0.17240769]
 [0.82770322 0.17229678]
 [0.88863678 0.11136322]
 [0.46885289 0.53114711]
 [0.90691939 0.09308061]
 [0.63940272 0.36059728]
 [0.80544541 0.19455459]
 [0.86452163 0.13547837]
 [0.8044931  0.1955069 ]
 [0.89125256 0.10874744]
 [0.8757965  0.1242035 ]
 [0.78740254 0.21259746]
 [0.67201813 0.32798187]
 [0.86101896 0.13898104]
 [0.78087078 0.21912922]
 [0.88161976 0.11838024]
 [0.86418429 0.13581571]
 [0.81339645 0.18660355]
 [0.89723545 0.10276455]
 [0.81702143 0.18297857]
 [0.85933888 0.14066112]
 [0.82703689 0.17296311]
 [0.8522979  0.1477021 ]
 [0.90303281 0.09696719]
 [0.89314949 0.10685051]
 [0.8814913  0.1185087 ]
 [0.79972534 0.20027466]
 [0.81745117 0.18254883]
 [0.85141647 0.14858353]
 [0.8064526  0.1935474 ]
 [0.80363771 0.19636229]
 [0.84361208 0.15638792]
 [0.87144157 0.12855843]
 [0.82321684 0.17678316]
 [0.29172642 0.70827358]
 [0.71686924 0.28313076]
 [0.61961262 0.38038738]
 [0.8293564  0.1706436 ]
 [0.42201855 0.57798145]
 [0.88855981 0.11144019]
 [0.68403487 0.31596513]
 [0.89486239 0.10513761]]
```

In [206]:

```
print(y_pred)
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2
 1 1 1]
```

In [ ]: