Import the Libraries

In [1]:

```
import numpy as np
import pandas as pd
```

Dataset Loading

In [2]:

```
data=pd.read_csv(r"C:\Users\Acer\Desktop\Dataset\heart_disease dataset.csv")
```

In [3]:

```
data
```

Out[3]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	

1025 rows × 14 columns

In [4]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [5]:

```
data.isnull().sum()
```

Out[5]:

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [6]:

```
data["target"].value_counts()
```

Out[6]:

```
1    526
0    499
Name: target, dtype: int64
```

Data Slicing

In [7]:

```
x=data.iloc[:,:-1].values
y=data.iloc[:,-1].values
```

Using Imbalancing Technique to Balance data

In [8]:

```
from imblearn.over_sampling import SMOTE
s=SMOTE()
x_data,y_data=s.fit_resample(x,y)
```

In [9]:

```
from collections import Counter
print("The Original data : ",Counter(y))
print("The Aritificial data : ",Counter(y_data))
```

```
The Original data :  Counter({1: 526, 0: 499})
The Aritificial data :  Counter({0: 526, 1: 526})
```

Cross Validation Technique

In [10]:

```
from sklearn.model_selection import KFold
kf=KFold(n_splits=6,random_state=5,shuffle=True)
kf.get_n_splits(x_data)
print(kf)
```

```
KFold(n_splits=6, random_state=5, shuffle=True)
```

In [11]:

```
for train_index,test_index in kf.split(x_data):
    print("TRAIN : ",train_index,"TEST : ",test_index)
    x_train,x_test=x_data[train_index],x_data[test_index]
    y_train,y_test=y_data[train_index],y_data[test_index]


    from sklearn.linear_model import LogisticRegression
    lr=LogisticRegression()
    from sklearn.model_selection import cross_val_score
    score=cross_val_score(lr,x_data,y_data,cv=kf)
    print(score)
    print(np.mean(score)*100)
```

```
TRAIN :  [   0    1    2    4    5    6    7    9   10   13   14   16   17
18
   19   20   21   22   24   25   26   27   29   30   31   32   36   37
   38   39   40   41   42   43   44   45   46   47   48   49   50   52
   53   54   55   56   57   58   61   62   63   64   65   66   67   68
   69   70   73   74   75   76   77   78   79   80   81   82   83   84
   87   88   89   90   91   92   93   94   95   96   97   98   99  100
  101  102  103  104  105  107  108  109  110  111  112  113  114  115
  116  117  118  119  120  122  123  124  125  126  127  128  129  130
  132  134  135  136  137  138  139  140  141  142  143  144  145  146
  147  149  150  151  152  153  154  155  156  157  158  160  161  163
  164  167  168  169  170  171  172  173  174  175  177  178  179  180
  181  182  183  184  185  186  187  188  189  190  191  192  194  195
  197  199  200  202  204  205  206  208  209  210  211  212  213  214
  215  216  217  218  220  221  223  224  225  226  227  229  231  232
  233  234  235  237  238  239  240  241  242  243  245  246  247  248
  249  251  252  253  255  256  257  259  260  261  262  263  264  265
  266  268  269  270  271  272  273  274  275  276  277  278  280  281
  282  284  285  287  288  289  290  291  292  294  295  296  297  298
```

Predicting the Test set results

In [12]:

```
from sklearn.model_selection import cross_val_predict
y_pred=cross_val_predict(lr,x_test,y_test,cv=kf)
print(y_pred)
```

```
C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(


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

C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
```

```
C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
C:\Users\Acer\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
```

Confusion Matrix , Accuracy score and Classification Report

In [13]:

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
ac = accuracy_score(y_pred,y_test)*100
cm = confusion_matrix(y_pred,y_test)
cr = classification_report(y_pred,y_test)
print("Accuracy score:",ac)
print("*******************************************")

print("Confusion matrix:")
print(cm)
print("*******************************************")

print("classification report:")
print(cr)
```

```
Accuracy score: 79.42857142857143
*******************************************
Confusion matrix:
[[57 14]
 [22 82]]
*******************************************
classification report:
              precision    recall  f1-score   support

           0       0.72      0.80      0.76        71
           1       0.85      0.79      0.82       104

    accuracy                           0.79       175
   macro avg       0.79      0.80      0.79       175
weighted avg       0.80      0.79      0.80       175
```

In [ ]: