In [1]:
```
#heart
import os
os.chdir('desktop')
```

In [2]:
```
import pandas as pd
import numpy as np
```

In [5]:
```
df=pd.read_csv("heart.csv")
```

In [6]:
```
df.head()
```

Out[6]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

In [ ]:
```
/* age (Age in years)

sex : (1 = male, 0 = female)

cp (Chest Pain Type): [ 0: asymptomatic, 1: atypical angina, 2:non-anginal pain, 3: typical angina]

trestbps (Resting Blood Pressure in mm/hg )

chol (Serum Cholesterol in mg/dl)

fps (Fasting Blood Sugar > 120 mg/dl): [0 = no, 1 = yes]

restecg (Resting ECG): [0: showing probable or definite left ventricular hypertrophy by Estes' criteria, 1: normal, 2: having ST-T

thalach (maximum heart rate achieved)

exang (Exercise Induced Angina): [1 = yes, 0 = no]
```

```
oldpeak (ST depression induced by exercise relative to rest)

slope (the slope of the peak exercise ST segment): [0: downsloping; 1: flat; 2: upsloping]

ca [number of major vessels (0-3)

thal : [1 = normal, 2 = fixed defect, 3 = reversible defect]

target: [0 = disease, 1 = no disease] */
```

In [8]:
```
#data cleaning
df.isnull().sum()
```

Out[8]:
```
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 [12]:
```
df.nunique()
```

Out[12]:
```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            5
thal          4
```

```
target       2
dtype: int64
```

In [10]:
```
#Error correction in thal and caa
df['ca'].unique()
```

Out[10]:
```
array([2, 0, 1, 3, 4], dtype=int64)
```

In [ ]:
```
#we have to replace all 4 by nan so that unique will be 4 and not 5
```

In [11]:
```
df['ca'].value_counts()
```

Out[11]:
```
0    578
1    226
2    134
3     69
4     18
Name: ca, dtype: int64
```

In [14]:
```
df.loc[df['ca']==4,'ca']=np.NaN
```

In [16]:
```
df.nunique()
```

Out[16]:
```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            4
thal          4
target        2
dtype: int64
```

In [20]:
```
#data modeling
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [26]:
```
df.isnull().sum()
```

Out[26]:
```
age          0
sex          0
cp           0
trestbps     0
chol         0
fbs          0
restecg      0
thalach      0
exang        0
oldpeak      0
slope        0
ca          18
thal         0
target       0
dtype: int64
```

In [29]:
```
df = df.fillna(df.median())
df.isnull().sum()
```

Out[29]:
```
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 [31]:
```
#splitting
x=df.drop(['target'],axis=1)
y=df['target']
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2, random_state=0)
```

In [32]:
```
#trainig
lg=LogisticRegression(random_state=0)
lg.fit(x_train,y_train)
```

```
y_train_pred=lg.predict(x_train)
print("Accuracy_training : ", accuracy_score(y_train_pred,y_train))
```

```
Accuracy_training :  0.8670731707317073
```

```
C:\Users\AKANKSHA\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
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [33]:
```
#testing
y_test_pred=lg.predict(x_test)
print("Accuracy_training : ", accuracy_score(y_test_pred,y_test))
```

```
Accuracy_training :  0.8731707317073171
```