Logistic Regression

In [3]:

```
#Name: Yash Pravin Gadbail
#Roll no. : 35
#Sec: 3rd A
#Sub : ET 1
#Date:05/10/2024
```

In [5]:

```
#Aim: To Perform Operation on Logistic Regression
```

In [7]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

In [1]:

```
import os
```

In [2]:

```
os.getcwd()
```

Out[2]:

```
'C:\\Users\\RAG'
```

In [20]:

```
os.chdir("C:\\Users\\OneDrive\\Desktop")
```

In [22]:

```
df=pd.read_csv("C:\\Users\\OneDrive\\Desktop\\framingham.csv")
```

In [24]:

```
df.head()
```

Out[24]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp
0	1	39	4.0	0	0.0	0.0	0	0
1	0	46	2.0	0	0.0	0.0	0	0
2	1	48	1.0	1	20.0	0.0	0	0
3	0	61	3.0	1	30.0	0.0	0	1
4	0	46	3.0	1	23.0	0.0	0	0

In [26]:

```
df.describe()
```

Out[26]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	pre
count	4240.000000	4240.000000	4135.000000	4240.000000	4211.000000	4187.000000	
mean	0.429245	49.580189	1.979444	0.494104	9.005937	0.029615	
std	0.495027	8.572942	1.019791	0.500024	11.922462	0.169544	
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000	
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000	
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000	
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000	
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000	

In [28]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4240 entries, 0 to 4239
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   male             4240 non-null   int64  
 1   age              4240 non-null   int64  
 2   education        4135 non-null   float64
 3   currentSmoker    4240 non-null   int64  
 4   cigsPerDay       4211 non-null   float64
 5   BPMeds           4187 non-null   float64
 6   prevalentStroke  4240 non-null   int64  
 7   prevalentHyp     4240 non-null   int64  
 8   diabetes         4240 non-null   int64  
 9   totChol          4190 non-null   float64
 10  sysBP            4240 non-null   float64
 11  diaBP            4240 non-null   float64
 12  BMI              4221 non-null   float64
 13  heartRate        4239 non-null   float64
 14  glucose          3852 non-null   float64
 15  TenYearCHD       4240 non-null   int64  
dtypes: float64(9), int64(7)
memory usage: 530.1 KB
```

In [30]:
```
df.isna().sum()
```

Out[30]:
```
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
dtype: int64
```

In [32]:
```
df
```

Out[32]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalent
0	1	39	4.0	0	0.0	0.0	0	
1	0	46	2.0	0	0.0	0.0	0	
2	1	48	1.0	1	20.0	0.0	0	
3	0	61	3.0	1	30.0	0.0	0	
4	0	46	3.0	1	23.0	0.0	0	
...	...	...	...	...	...	...	...	
4235	0	48	2.0	1	20.0	NaN	0	
4236	0	44	1.0	1	15.0	0.0	0	
4237	0	52	2.0	0	0.0	0.0	0	
4238	1	40	3.0	0	0.0	0.0	0	
4239	0	39	3.0	1	30.0	0.0	0	

4240 rows × 16 columns

# Missing Value Treatment

In [35]:
```
df['glucose'].fillna(value = df['glucose'].mean(),inplace=True)
```

In [37]:
```
df['education'].fillna(value = df['education'].mean(),inplace=True)
```

In [52]:
```
df['heartRate'].fillna(value = df['heartRate'].mean(),inplace=True)
```

In [41]:
```
df['BMI'].fillna(value = df['BMI'].mean(),inplace=True)
```

In [45]:
```
df['cigsPerDay'].fillna(value = df['cigsPerDay'].mean(),inplace=True)
```

In [48]:
```
df['totChol'].fillna(value = df['totChol'].mean(),inplace=True)
```

In [50]:
```
df['BPMeds'].fillna(value = df['BPMeds'].mean(),inplace=True)
```

In [54]:
```
df.isna().sum()
```

Out[54]:
```
male               0
age                0
education          0
currentSmoker      0
cigsPerDay         0
BPMeds             0
prevalentStroke    0
prevalentHyp       0
diabetes           0
totChol            0
sysBP              0
diaBP              0
BMI                0
heartRate          0
glucose            0
TenYearCHD         0
dtype: int64
```

In [56]:
```
#Splitting the dependent and independent variables.
x = df.drop("TenYearCHD",axis=1)
y = df['TenYearCHD']
```

In [60]:
```
x
```

Out[60]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalent
0	1	39	4.0	0	0.0	0.000000	0	
1	0	46	2.0	0	0.0	0.000000	0	
2	1	48	1.0	1	20.0	0.000000	0	
3	0	61	3.0	1	30.0	0.000000	0	
4	0	46	3.0	1	23.0	0.000000	0	
...	...	...	...	...	...	...	...	
4235	0	48	2.0	1	20.0	0.029615	0	
4236	0	44	1.0	1	15.0	0.000000	0	
4237	0	52	2.0	0	0.0	0.000000	0	
4238	1	40	3.0	0	0.0	0.000000	0	
4239	0	39	3.0	1	30.0	0.000000	0	

4240 rows × 15 columns

Train Test Split

```
In [63]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,train_si
```

```
In [65]: y_train
```

```
Out[65]: 2399    0
         3609    0
         3814    0
         1941    0
         4021    0
         1878    1
         93      0
         4221    1
         3095    0
         1792    0
         966     0
         2568    0
         3300    0
         818     0
         1541    0
         331     0
         1394    0
         583     0
         866     0
         3148    1
         2890    0
         1884    0
         1279    0
         2581    0
         2326    0
         1999    0
         2133    0
         4032    0
         4168    0
         1801    0
         3874    1
         1709    0
         3126    0
         2664    0
         2616    0
         2253    0
         1147    0
         1794    0
         42      1
         3538    0
         1398    0
         2417    0
         Name: TenYearCHD, dtype: int64
```

Logistic Regression Algorithm

In [68]:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression().fit(x_train,y_train)
model.score(x_train, y_train)
```

Out[68]:

```
0.9047619047619048
```

In [ ]: