Build a logistic regression model to predict Diabetes

In [1]:
```
#Load the datset from sklearn
import pandas as pd
import numpy as np
from sklearn.datasets import load_diabetes

data=load_diabetes()
```

In [2]:
```
data
```

Out[2]:
```
{'data': array([[ 0.03807591,  0.05068012,  0.06169621, ..., -0.00259226,
          0.01990842, -0.01764613],
        [-0.00188202, -0.04464164, -0.05147406, ..., -0.03949338,
         -0.06832974, -0.09220405],
        [ 0.08529891,  0.05068012,  0.04445121, ..., -0.00259226,
          0.00286377, -0.02593034],
        ...,
        [ 0.04170844,  0.05068012, -0.01590626, ..., -0.01107952,
         -0.04687948,  0.01549073],
        [-0.04547248, -0.04464164,  0.03906215, ...,  0.02655962,
          0.04452837, -0.02593034],
        [-0.04547248, -0.04464164, -0.0730303 , ..., -0.03949338,
         -0.00421986,  0.00306441]]),
 'target': array([151.,  75., 141., 206., 135.,  97., 138.,  63., 110., 310., 10
1.,
         69., 179., 185., 118., 171., 166., 144.,  97., 168.,  68.,  49.,
         68., 245., 184., 202., 137.,  85., 131., 283., 129.,  59., 341.,
         87.,  65., 102., 265., 276., 252.,  90., 100.,  55.,  61.,  92.,
        259.,  53., 190., 142.,  75., 142., 155., 225.,  59., 104., 182.,
        128.,  52.,  37., 170., 170.,  61., 144.,  52., 128.,  71., 163.,
        150.,  97., 160., 178.,  48., 270., 202., 111.,  85.,  42., 170.,
        200., 252., 113., 143.,  51.,  52., 210.,  65., 141.,  55., 134.,
         42., 111.,  98., 164.,  48.,  96.,  90., 162., 150., 279.,  92.,
         83., 128., 102., 302., 198.,  95.,  53., 134., 144., 232.,  81.,
        104.,  59., 246., 297., 258., 229., 275., 281., 179., 200., 200.,
        173., 180.,  84., 121., 161.,  99., 109., 115., 268., 274., 158.,
        107.,  83., 103., 272.,  85., 280., 336., 281., 118., 317., 235.,
         60., 174., 259., 178., 128.,  96., 126., 288.,  88., 292.,  71.,
        197., 186.,  25.,  84.,  96., 195.,  53., 217., 172., 131., 214.,
         59.,  70., 220., 268., 152.,  47.,  74., 295., 101., 151., 127.,
        237., 225.,  81., 151., 107.,  64., 138., 185., 265., 101., 137.,
        143., 141.,  79., 292., 178.,  91., 116.,  86., 122.,  72., 129.,
        142.,  90., 158.,  39., 196., 222., 277.,  99., 196., 202., 155.,
         77., 191.,  70.,  73.,  49.,  65., 263., 248., 296., 214., 185.,
         78.,  93., 252., 150.,  77., 208.,  77., 108., 160.,  53., 220.,
        154., 259.,  90., 246., 124.,  67.,  72., 257., 262., 275., 177.,
         71.,  47., 187., 125.,  78.,  51., 258., 215., 303., 243.,  91.,
        150., 310., 153., 346.,  63.,  89.,  50.,  39., 103., 308., 116.,
        145.,  74.,  45., 115., 264.,  87., 202., 127., 182., 241.,  66.,
         94., 283.,  64., 102., 200., 265.,  94., 230., 181., 156., 233.,
         60., 219.,  80.,  68., 332., 248.,  84., 200.,  55.,  85.,  89.,
         31., 129.,  83., 275.,  65., 198., 236., 253., 124.,  44., 172.,
        114., 142., 109., 180., 144., 163., 147.,  97., 220., 190., 109.,
        191., 122., 230., 242., 248., 249., 192., 131., 237.,  78., 135.,
        244., 199., 270., 164.,  72.,  96., 306.,  91., 214.,  95., 216.,
        263., 178., 113., 200., 139., 139.,  88., 148.,  88., 243.,  71.,
         77., 109., 272.,  60.,  54., 221.,  90., 311., 281., 182., 321.,
         58., 262., 206., 233., 242., 123., 167.,  63., 197.,  71., 168.,
        140., 217., 121., 235., 245.,  40.,  52., 104., 132.,  88.,  69.,
        219.,  72., 201., 110.,  51., 277.,  63., 118.,  69., 273., 258.,
         43., 198., 242., 232., 175.,  93., 168., 275., 293., 281.,  72.,
        140., 189., 181., 209., 136., 261., 113., 131., 174., 257.,  55.,
         84.,  42., 146., 212., 233.,  91., 111., 152., 120.,  67., 310.,
         94., 183.,  66., 173.,  72.,  49.,  64.,  48., 178., 104., 132.,
        220.,  57.]),
 'frame': None,
 'DESCR': '.. _diabetes_dataset:\n\nDiabetes dataset\n----------------\n\nTen bas
eline variables, age, sex, body mass index, average blood\npressure, and six bloo
```

```
d serum measurements were obtained for each of n =\n442 diabetes patients, as wel
l as the response of interest, a\nquantitative measure of disease progression one
year after baseline.\n\n**Data Set Characteristics:**\n\n  :Number of Instances:
442\n\n  :Number of Attributes: First 10 columns are numeric predictive values\n
\n  :Target: Column 11 is a quantitative measure of disease progression one year
after baseline\n\n  :Attribute Information:\n      - age     age in years\n
- sex\n      - bmi     body mass index\n      - bp      average blood pressure\n
- s1      tc, total serum cholesterol\n      - s2      ldl, low-density lipoprote
ins\n      - s3      hdl, high-density lipoproteins\n      - s4      tch, total c
holesterol / HDL\n      - s5      ltg, possibly log of serum triglycerides level
\n      - s6      glu, blood sugar level\n\nNote: Each of these 10 feature variab
les have been mean centered and scaled by the standard deviation times `n_samples
` (i.e. the sum of squares of each column totals 1).\n\nSource URL:\nhttps://www
4.stat.ncsu.edu/~boos/var.select/diabetes.html\n\nFor more information see:\nBrad
ley Efron, Trevor Hastie, Iain Johnstone and Robert Tibshirani (2004) "Least Angl
e Regression," Annals of Statistics (with discussion), 407-499.\n(https://web.sta
nford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf)',
 'feature_names': ['age',
  'sex',
  'bmi',
  'bp',
  's1',
  's2',
  's3',
  's4',
  's5',
  's6'],
 'data_filename': 'C:\\ProgramData\\Anaconda3\\lib\\site-packages\\sklearn\\datas
ets\\data\\diabetes_data.csv.gz',
 'target_filename': 'C:\\ProgramData\\Anaconda3\\lib\\site-packages\\sklearn\\dat
asets\\data\\diabetes_target.csv.gz'}
```

In [3]:
```
data.feature_names
```

Out[3]:
```
['age', 'sex', 'bmi', 'bp', 's1', 's2', 's3', 's4', 's5', 's6']
```

In [4]:
```
data.DESCR
```

Out[4]:
```
'.. _diabetes_dataset:\n\nDiabetes dataset\n----------------\n\nTen baseline vari
ables, age, sex, body mass index, average blood\npressure, and six blood serum me
asurements were obtained for each of n =\n442 diabetes patients, as well as the r
esponse of interest, a\nquantitative measure of disease progression one year afte
r baseline.\n\n**Data Set Characteristics:**\n\n  :Number of Instances: 442\n\n
:Number of Attributes: First 10 columns are numeric predictive values\n\n  :Targe
t: Column 11 is a quantitative measure of disease progression one year after base
line\n\n  :Attribute Information:\n      - age     age in years\n      - sex\n
- bmi     body mass index\n      - bp      average blood pressure\n      - s1
tc, total serum cholesterol\n      - s2      ldl, low-density lipoproteins\n
- s3      hdl, high-density lipoproteins\n      - s4      tch, total cholesterol
/ HDL\n      - s5      ltg, possibly log of serum triglycerides level\n      - s6
glu, blood sugar level\n\nNote: Each of these 10 feature variables have been mean
centered and scaled by the standard deviation times `n_samples` (i.e. the sum of
squares of each column totals 1).\n\nSource URL:\nhttps://www4.stat.ncsu.edu/~boo
s/var.select/diabetes.html\n\nFor more information see:\nBradley Efron, Trevor Ha
stie, Iain Johnstone and Robert Tibshirani (2004) "Least Angle Regression," Annal
s of Statistics (with discussion), 407-499.\n(https://web.stanford.edu/~hastie/Pa
pers/LARS/LeastAngle_2002.pdf)'
```

In [5]:
```
#load diabetes.csv
diab=pd.read_csv("C:/1562_AIML/diabetes.csv")
```

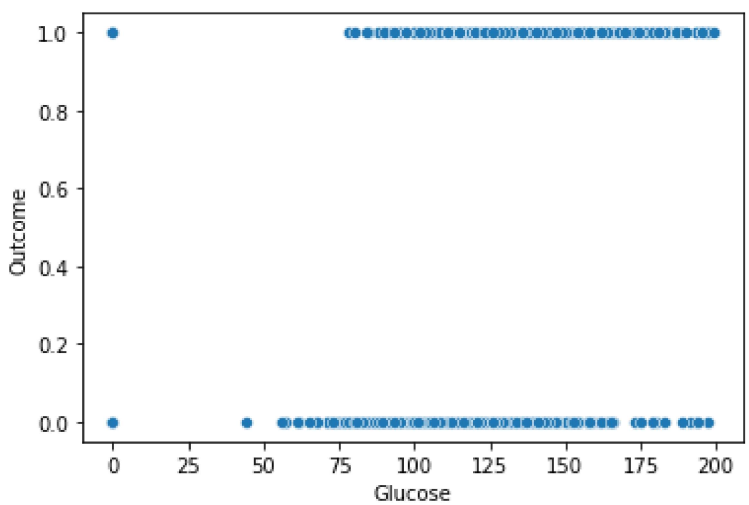
In [7]:
```
diab.head()
```

Out[7]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148	72	35	0	33.6	0.627	50
1	1	85	66	29	0	26.6	0.351	31
2	8	183	64	0	0	23.3	0.672	32
3	1	89	66	23	94	28.1	0.167	21
4	0	137	40	35	168	43.1	2.288	33

In [14]:
```
#draw a scatter plot of glucose level against outcome
import seaborn as sns
sns.scatterplot(data=diab,x='Glucose', y='Outcome')
```

Out[14]: <AxesSubplot:xlabel='Glucose', ylabel='Outcome'>



In [10]:
```
diab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [15]:
```
#prepare x and y
features=['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI',
x=diab[features]
y=diab.Outcome
```

In [17]:
```
#split the dataset in training and test dataset
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.8,random_state=100
```

In [23]:
```
#Build logistics regression model
from sklearn.linear_model import LogisticRegression
logr=LogisticRegression(max_iter=1000)

#train the model
logr.fit(x_train,y_train)

#test the model
y_pred=logr.predict(x_test)
```

In [25]:
```
#compute the accuracy of the model
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix

print("Accuracy score:",accuracy_score(y_test,y_pred))
```

```
Accuracy score: 0.7402597402597403
```

In [27]:
```
#print classification report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.77      0.85      0.81       101
           1       0.65      0.53      0.58        53

    accuracy                           0.74       154
   macro avg       0.71      0.69      0.70       154
weighted avg       0.73      0.74      0.73       154
```

In [30]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[86 15]
 [25 28]]
```

In [ ]: