In [1]:

```
import numpy as np
import pandas as pd
df = pd.read_csv("diabetes.csv")
df
```

Out[1]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

In [11]:

```
zero_not_accepted = ["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]
for i in zero_not_accepted:
    df[i] = df[i].replace(0,np.NaN)
    mean = int(df[i].mean(skipna=True))
    df[i] = df[i].replace(np.NaN,mean)
```

In [12]:

```
df
```

Out[12]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35.0	155.0	33.6	0.627	50	1

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
1	1	85.0	66.0	29.0	155.0	26.6	0.351	31	0
2	8	183.0	64.0	29.0	155.0	23.3	0.672	32	1
3	1	89.0	66.0	23.0	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35.0	168.0	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101.0	76.0	48.0	180.0	32.9	0.171	63	0
764	2	122.0	70.0	27.0	155.0	36.8	0.340	27	0
765	5	121.0	72.0	23.0	112.0	26.2	0.245	30	0
766	1	126.0	60.0	29.0	155.0	30.1	0.349	47	1
767	1	93.0	70.0	31.0	155.0	30.4	0.315	23	0

768 rows × 9 columns

In [13]:

```
df.isnull().sum()
```

Out[13]:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [16]:

```
x = df.iloc[:,:8]
y = df.iloc[:,8]
x
```

Out[16]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	6	148.0	72.0	35.0	155.0	33.6	0.627	50

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
1	1	85.0	66.0	29.0	155.0	26.6	0.351	31
2	8	183.0	64.0	29.0	155.0	23.3	0.672	32
3	1	89.0	66.0	23.0	94.0	28.1	0.167	21
4	0	137.0	40.0	35.0	168.0	43.1	2.288	33
...	...	...	...	...	...	...	...	...
763	10	101.0	76.0	48.0	180.0	32.9	0.171	63
764	2	122.0	70.0	27.0	155.0	36.8	0.340	27
765	5	121.0	72.0	23.0	112.0	26.2	0.245	30
766	1	126.0	60.0	29.0	155.0	30.1	0.349	47
767	1	93.0	70.0	31.0	155.0	30.4	0.315	23

768 rows × 8 columns

In [32]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.35,random_state=2)
x_train.shape,x_test.shape
```

Out[32]:

```
((499, 8), (269, 8))
```

In [33]:

```
from sklearn.preprocessing import StandardScaler
scale = StandardScaler()
x_train = scale.fit_transform(x_train)
x_test = scale.fit_transform(x_test)
x_train
```

Out[33]:

```
array([[-0.27184134,  0.88007084,  0.28281852, ..., -1.61011059,
        -0.8033334 ,  0.32264881],
       [ 1.48602688, -0.03743026, -1.37456774, ...,  0.14566373,
         1.8135394 , -0.02192424],
       [-0.85779742, -0.52894871, -1.87178361, ..., -0.56806567,
        -0.87834849, -0.36649729],
       ...,
       [ 0.02113669,  0.06087343, -0.21439736, ..., -0.48241814,
         1.90009528,  1.01179491],
       [-0.27184134, -0.23403764,  0.11707989, ..., -0.85355743,
        -1.091853  , -0.7972136 ],
       [ 0.02113669, -0.43064502, -0.54587461, ..., -0.0541805 ,
        -0.04164165, -0.36649729]])
```

In [34]:
```
k = np.sqrt(len(x_test))
k
```

Out[34]:
```
16.401219466856727
```

In [35]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=11)
knn.fit(x_train,y_train)
```

Out[35]:
```
KNeighborsClassifier(n_neighbors=11)
```

In [36]:
```
y_pred = knn.predict(x_test)
y_pred[0:5]
```

Out[36]:
```
array([0, 0, 0, 1, 0], dtype=int64)
```

In [38]:
```
from sklearn.metrics import accuracy_score, confusion_matrix
accuracy_score(y_test,y_pred)*100
```

Out[38]:
```
72.86245353159852
```

In [39]:
```
confusion_matrix(y_test,y_pred)
```

Out[39]:
```
array([[148,  36],
       [ 37,  48]], dtype=int64)
```

In [40]:
```
y_pred_probab = knn.predict_proba(x_test[0:5])
y_pred_probab
```

Out[40]:
```
array([[1.        , 0.        ],
       [0.72727273, 0.27272727],
       [0.90909091, 0.09090909],
       [0.36363636, 0.63636364],
       [0.72727273, 0.27272727]])
```

In [ ]: