In [1]:

```
# import libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
```

In [21]:

```
# Read csv file
la = pd.read_csv('diabetes.csv')
print ( len (la))
print(la.head())
```

```
768
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

In [ ]:

```
# Blood pressure and Glucose does not accept the zero Because it can affect the outcome
#  np stand for Numpy and NaN stand for (does not exists )
#  Skipna is used to skip values while mean is calculated
```

In [9]:

```
Zero_not_accepted_Due_to_null_values = ['Glucose', 'BloodPressure', 'SkinThickness', 'BMI',
for column in Zero_not_accepted_Due_to_null_values:
    la[column] = la[column].replace(0, np.NaN)
    mean = int(la[column].mean(skipna=True))
    la[column] = la[column].replace(np.NaN, mean)
```

In [22]:

```
print(la['Glucose'])
```

```
0      148
1       85
2      183
3       89
4      137
      ...
763    101
764    122
765    121
766    126
767     93
Name: Glucose, Length: 768, dtype: int64
```

In [ ]:

```
#  Before any other process we need to split the data in traning testing and splitting
#  iloc returns a Pandas Series when one row is selected, and a Pandas DataFrame when multi
#  First row and second is column
```

In [10]:

```
X = la.iloc[:, 0:8]
y = la.iloc[:, 8]
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0, test_size=0.2)
```

In [11]:

```
print(len(X_train))
print(len(y_train))
print(len(X_test))
print(len(y_test))
```

```
614
614
154
154
```

In [12]:

```
# Feature Scalling
# import StandardScalar
# X_fit_transform is used to fit the variable in (x_train)
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

In [13]:

```
#  Classifer is used to fit the train data in model
classifier = KNeighborsClassifier(n_neighbors=11, p=2,metric='euclidean')
```

In [15]:

```
classifier.fit(X_train, y_train)
```

Out[15]:

```
KNeighborsClassifier(metric='euclidean', n_neighbors=11)
```

In [16]:

```
y_pred = classifier.predict(X_test)
y_pred
```

Out[16]:

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

In [17]:

```
#  Evaluate the model
#  f1 is used to Count the both side of balanced
cm = confusion_matrix(y_test, y_pred)
print (cm)
print(f1_score(y_test, y_pred))
```

```
[[94 13]
 [15 32]]
0.6956521739130436
```

In [18]:

```
print(accuracy_score(y_test, y_pred))
```

```
0.8181818181818182
```

In [ ]: