In [1]:
```
    import numpy as np
    import pandas as pd
```

In [2]:
```
data = pd.read_csv('./diabetes.csv')
data.head()
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	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 [3]:
```
#Check for null or missing values
data.isnull().sum()
```

Out[3]:
```
Pregnancies      0
Glucose          0
BloodPressure    0
SkinThickness    0
Insulin          0
BMI              0
Pedigree         0
Age              0
Outcome          0
dtype: int64
```

In [4]:
```
#Replace zero values with mean values
for column in data.columns[1:-3]:
    data[column].replace(0, np.NaN, inplace = True)
    data[column].fillna(round(data[column].mean(skipna=True)), inplace
data.head(10)
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age
0	6	148.0	72.0	35.0	156.0	33.6	0.627	50
1	1	85.0	66.0	29.0	156.0	26.6	0.351	31
2	8	183.0	64.0	29.0	156.0	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
5	5	116.0	74.0	29.0	156.0	25.6	0.201	30
6	3	78.0	50.0	32.0	88.0	31.0	0.248	26
7	10	115.0	72.0	29.0	156.0	35.3	0.134	29
8	2	197.0	70.0	45.0	543.0	30.5	0.158	53
9	8	125.0	96.0	29.0	156.0	32.0	0.232	54

In [5]:
```
X = data.iloc[:, :8] #Features
Y = data.iloc[:, 8:] #Predictor
```

In [6]:
```
#Perform Spliting
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2
```

In [7]:
```
#KNN
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn_fit = knn.fit(X_train, Y_train.values.ravel())
knn_pred = knn_fit.predict(X_test)
```

In [10]:
```
from sklearn.metrics import confusion_matrix, precision_score, recall_s
print("Confusion Matrix")
print(confusion_matrix(Y_test, knn_pred))
print("Accuracy Score:", accuracy_score(Y_test, knn_pred))
print("Reacal Score:", recall_score(Y_test, knn_pred))
print("F1 Score:", f1_score(Y_test, knn_pred))
print("Precision Score:",precision_score(Y_test, knn_pred))
print("Error Rate:",(1-accuracy_score(Y_test, knn_pred)))
```

```
Confusion Matrix
[[88 19]
 [19 28]]
Accuracy Score: 0.7532467532467533
Reacal Score: 0.5957446808510638
F1 Score: 0.5957446808510638
Precision Score: 0.5957446808510638
Error Rate: 0.24675324675324672
```