In [2]:

```
#importing libraries
import pandas as pd
import numpy as np
```

In [27]:

```
#importing dataset
dt= pd.read_csv("diabities.csv")
dt.head(10)
```

Out[27]:

| | Pregnancies | Glucose | blood pressure | skin thickness | Insulin | BMI | DiabetesPedigreeFunction | Age | Outcome |
|---|---|---|---|---|---|---|---|---|---|
| **NaN** | 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 50 | 1 |
| **NaN** | 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 31 | 0 |
| **NaN** | 8 | 183 | 64 | 0 | 0 | 23.3 | 0.672 | 32 | 1 |
| **NaN** | 1 | 89 | 66 | 23 | 94 | 28.1 | 0.167 | 21 | 0 |
| **NaN** | 0 | 137 | 40 | 35 | 168 | 43.1 | 2.288 | 33 | 1 |
| **NaN** | 5 | 116 | 74 | 0 | 0 | 25.6 | 0.201 | 30 | 0 |
| **NaN** | 3 | 78 | 50 | 32 | 88 | 31.0 | 0.248 | 26 | 1 |
| **NaN** | 10 | 115 | 0 | 0 | 0 | 35.3 | 0.134 | 29 | 0 |
| **NaN** | 2 | 197 | 70 | 45 | 543 | 30.5 | 0.158 | 53 | 1 |
| **NaN** | 8 | 125 | 96 | 0 | 0 | 0.0 | 0.232 | 54 | 1 |

In [28]:

```
dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Float64Index: 768 entries, nan to nan
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   blood pressure            768 non-null    int64  
 3   skin thickness            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: 60.0 KB
```

In [30]:

```
dt.isnull().sum()
```

Out[30]:

```
Pregnancies                 0
Glucose                     0
blood pressure              0
skin thickness              0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [34]:

```
#assigning independent and dependenet variables
X= dt.iloc[:,:-1]
Y= dt.iloc[:,-1]
```

In [35]:

```
#spilliting data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test= train_test_split(X,Y,test_size=25, random_state=0)
```

In [36]:

```
#applying classification algorithms
```

In [37]:

```
#Random Forest
from sklearn.ensemble import RandomForestClassifier
classifier= RandomForestClassifier(n_estimators=6,criterion='entropy',random_state=0)
classifier.fit(x_train,y_train)

y_pred=classifier.predict(x_test)
```

In [45]:

```
from sklearn.metrics import accuracy_score
acc= round(accuracy_score(y_pred,y_test),2)*100
acc
```

Out[45]:

```
88.0
```

In [47]:

```
#Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,r2_score,classification_report
loreg=LogisticRegression(solver='lbfgs',max_iter=1000)
loreg.fit(x_train,y_train)
y_pred=loreg.predict(x_test)
acc_loreg= round(accuracy_score(y_pred,y_test),2)*100
acc_loreg
```

Out[47]:

```
96.0
```

In [50]:

```
#KNN
from sklearn.neighbors import KNeighborsClassifier
knn= KNeighborsClassifier(n_neighbors=2)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
acc_knn= round(accuracy_score(y_pred,y_test),2)*100
acc_knn
```

Out[50]:

```
80.0
```

In [ ]: