In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
dataset = pd.read_csv("Diabities.csv")
dataset.head(5)
```

Out[2]:

	Pregnancies	Glucose	blood pressure	skin thickness	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

In [3]:

```
dataset.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   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: 54.1 KB
```

In [4]:

```
dataset.isnull().sum()
```

Out[4]:

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

In [5]:

```
x = dataset.iloc[:,:-1]
y = dataset.iloc[:,-1]
```

In [6]:

```
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 [7]:

```
#random forest
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=8, criterion= 'entropy', random_state=0
)
classifier.fit(x_train,y_train)
y_pred = classifier.predict(x_test)
```

In [8]:

```
from sklearn.metrics import accuracy_score
acc_logreg2 = round(accuracy_score(y_pred, y_test), 2)*100
print("Accuracy: ",acc_logreg2)
```

```
Accuracy:  88.0
```

In [9]:

```
x
```

Out[9]:

	Pregnancies	Glucose	blood pressure	skin thickness	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
...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63
764	2	122	70	27	0	36.8	0.340	27
765	5	121	72	23	112	26.2	0.245	30
766	1	126	60	0	0	30.1	0.349	47
767	1	93	70	31	0	30.4	0.315	23

768 rows × 8 columns

In [10]:

```
x_train
```

Out[10]:

	Pregnancies	Glucose	blood pressure	skin thickness	Insulin	BMI	DiabetesPedigreeFunction	Age
605	1	124	60	32	0	35.8	0.514	21
239	0	104	76	0	0	18.4	0.582	27
744	13	153	88	37	140	40.6	1.174	39
79	2	112	66	22	0	25.0	0.307	24
496	5	110	68	0	0	26.0	0.292	30
...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63
192	7	159	66	0	0	30.4	0.383	36
629	4	94	65	22	0	24.7	0.148	21

	Pregnancies	Glucose	blood pressure	skin thickness	Insulin	BMI	DiabetesPedigreeFunction	Age
559	11	85	74	0	0	30.1	0.300	35
684	5	136	82	0	0	0.0	0.640	69

743 rows × 8 columns

In [11]:

```
y_train
```

Out[11]:

```
605    0
239    0
744    0
79     0
496    0
      ..
763    0
192    1
629    0
559    0
684    0
Name: Outcome, Length: 743, dtype: int64
```

In [12]:

```
#K Nearest Neighbor
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
print("Accuracy: ",acc_knn)
```

```
Accuracy:  80.0
```

In [13]:

```
#Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score,r2_score,classification_report
logreg = LogisticRegression(solver='lbfgs',max_iter=1000)
logreg.fit(x_train, y_train)
y_pred = logreg.predict(x_test)
acc_logreg1 = round(accuracy_score(y_pred, y_test), 2)*100
print("Accuracy: ",acc_logreg1)
```

```
Accuracy:  96.0
```

In [ ]:

```

```