In [1]:
```
#Aim:To perform Data Manipulation and using Pandas i.e. (Feature selection)
```

In [ ]:
```
#Name: Achal Subhash Kharwade
#Roll No: 36
#Sec: B
#Date :05-08-2023
```

In [3]:
```
# Theory: Theory:-Data manipulation is the method of organizing data to
# make it easier to read or more designed or structured.For
# instance a collection of any kind of data could be organized in
# alphabetical order so that it can be understood easily.
```

In [4]:
```
import pandas as pd
import os
os.getcwd()
```

Out[4]:
```
'C:\\Users\\Lenovo\\DSS 5th Sem'
```

In [21]:
```
os.chdir("D:\DSS\DSS PRAC PG")
```

In [6]:
```
data=pd.read_csv("diabetes.csv")
```

In [7]:
```
data.head()
```

Out[7]:

	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

In [8]:
```
#Pandas.size,.shape and .ndim are used to return size, shape and dimensions of data
#Returns tuple of shape (Rows,columns) of data
```

In [9]:
```
data.shape
```

Out[9]:
```
(768, 9)
```

In [10]:
```
data.size
```

Out[10]:
```
6912
```

In [11]:
```
data.ndim
```

Out[11]:
```
2
```

In [14]:
```
data.columns
```

Out[14]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [15]:
```
data.head()
```

Out[15]:

	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

In [16]:
```
data.drop(labels="Age",axis=1)
```

Out[16]:

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

768 rows × 8 columns

In [17]:
```
data.drop(labels=["Age","Glucose"],axis=1)
```

Out[17]:

	Pregnancies	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Outcome
0	6	72	35	0	33.6	0.627	1
1	1	66	29	0	26.6	0.351	0
2	8	64	0	0	23.3	0.672	1
3	1	66	23	94	28.1	0.167	0
4	0	40	35	168	43.1	2.288	1
...	...	...	...	...	...	...	...
763	10	76	48	180	32.9	0.171	0
764	2	70	27	0	36.8	0.340	0
765	5	72	23	112	26.2	0.245	0
766	1	60	0	0	30.1	0.349	1
767	1	70	31	0	30.4	0.315	0

768 rows × 7 columns

In [18]:
```
data.drop(labels=2,axis=0)
```

Out[18]:

	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
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
...	...	...	...	...	...	...	...	...	...
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

767 rows × 9 columns

In [19]:
```
data.head(10)
```

Out[19]:

	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
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

In [20]:
```
data.drop(labels=[2,3],axis=0)
```

Out[20]:

	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
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	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

766 rows × 9 columns