In [1]:
```
#Exp No : 2
```

In [2]:
```
#Aim: To perform Data Manipulation using pandas i.e. (Feature Selection)
```

In [3]:
```
#Name : Kaushal A. Bharade
#Roll No : 11
#Sec : A
#Subject : Data Science and Statistics
#Date : 05/08/2023
```

In [4]:
```
#import the basic library
import pandas as pd
```

In [5]:
```
import os
```

In [6]:
```
os.getcwd()
```

Out[6]:
```
'C:\\Users\\HP'
```

In [7]:
```
os.chdir("C:\\Users\\HP\\Desktop")
```

In [8]:
```
data=pd.read_csv("diabetes.csv")
```

In [9]:
```
data.head(10)
```

Out[9]:

	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 [10]:
```
data.tail()
```

Out[10]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
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

In [11]:
```
#Pandas,size,.shape and .ndim are used to return size,shape and dimension of data
#Return tuple of shape(Rows,Column) of data
```

In [12]:
```
data.shape
#returns size of dataframe/series which s equivalent to total number of elements.
#That is rows x columns.
```

Out[12]:
```
(768, 9)
```

In [13]:
```
data.size
```

Out[13]:
```
6912
```

In [14]:
```
#returns dimension of dataframe/series. 1 for one dimension(series), 2 for two dimension
data.ndim
```

Out[14]:
```
2
```

In [15]:
```
data.columns
```

Out[15]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [16]:
```
data.head
```
Out[16]:
```
<bound method NDFrame.head of      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
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     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
..                        ...  ...      ...
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[768 rows x 9 columns]>
```
In [17]:
```
#Drop is used to Drop one or more than one column from a data
#axis=1 i.e. Column
data.drop(labels="Age",axis=1)
```
Out[17]:

	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 [18]:
```
data.drop(labels=["Age","Glucose"],axis=1)
```
Out[18]:

	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 [19]:
```
data.drop(labels=2,axis=0)
```

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
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 [20]:

```
data.head(15)
```

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
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
10	4	110	92	0	0	37.6	0.191	30	0
11	10	168	74	0	0	38.0	0.537	34	1
12	10	139	80	0	0	27.1	1.441	57	0
13	1	189	60	23	846	30.1	0.398	59	1
14	5	166	72	19	175	25.8	0.587	51	1

In [21]:

```
#Drop is used to drop one or more than one column from a data
#axis=0 i.e. Row
data.drop(labels=[2,3],axis=0)
```

Out[21]:

	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

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js