In [1]:
```
#Exp No : 6
```

In [2]:
```
#Aim : To perform Simple Linear Regression and find out the coefficients of it.
```

In [3]:
```
# Name : Rohit Dadgal
# Roll no : 18
# Sec: A
# Subject : Data Science
#Date : 30/09/2023
```

In [4]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

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

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

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

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

In [8]:
```
df=pd.read_csv("Salary_dataset.csv")
```

In [9]:
```
df.head()
```

Out[9]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [10]:
```
df.head(10)
```

Out[10]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  35 non-null     float64
 1   Salary           35 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 692.0 bytes
```

In [12]:
```
df.tail()
```

Out[12]:

	YearsExperience	Salary
30	11.2	127345
31	11.5	126756
32	12.3	128765
33	12.9	135675
34	13.5	139465

In [13]:

```
df.describe()
```

Out[13]:

	YearsExperience	Salary
count	35.000000	35.000000
mean	6.308571	83945.600000
std	3.618610	32162.673003
min	1.100000	37731.000000
25%	3.450000	57019.000000
50%	5.300000	81363.000000
75%	9.250000	113223.500000
max	13.500000	139465.000000

In [14]:

```
df.shape
```

Out[14]:

```
(35, 2)
```

In [15]:

```
df.size
```

Out[15]:

```
70
```

In [16]:

```
df.ndim
```

Out[16]:

```
2
```

In [17]:

```
df.isnull()
```

Out[17]:

	YearsExperience	Salary
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
5	False	False
6	False	False
7	False	False
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False
13	False	False
14	False	False
15	False	False
16	False	False
17	False	False
18	False	False
19	False	False
20	False	False
21	False	False
22	False	False
23	False	False
24	False	False
25	False	False
26	False	False
27	False	False
28	False	False
29	False	False
30	False	False
31	False	False
32	False	False
33	False	False
34	False	False

In [18]:

```
df.isnull()
```

Out[18]:

	YearsExperience	Salary
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
5	False	False
6	False	False
7	False	False
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False
13	False	False
14	False	False
15	False	False
16	False	False
17	False	False
18	False	False
19	False	False
20	False	False
21	False	False
22	False	False
23	False	False
24	False	False
25	False	False
26	False	False
27	False	False
28	False	False
29	False	False
30	False	False
31	False	False
32	False	False
33	False	False
34	False	False

In [19]:

```
df.isnull
```

Out[19]:

```
<bound method DataFrame.isnull of     YearsExperience  Salary
0               1.1   39343
1               1.3   46205
2               1.5   37731
3               2.0   43525
4               2.2   39891
5               2.9   56642
6               3.0   60150
7               3.2   54445
8               3.2   64445
9               3.7   57189
10              3.9   63218
11              4.0   55794
12              4.0   56957
13              4.1   57081
14              4.5   61111
15              4.9   67938
16              5.1   66029
17              5.3   83088
18              5.9   81363
19              6.0   93940
20              6.8   91738
21              7.1   98273
22              7.9  101302
23              8.2  113812
24              8.7  109431
25              9.0  105582
26              9.5  116969
27              9.6  112635
28             10.3  122391
29             10.5  121872
30             11.2  127345
31             11.5  126756
32             12.3  128765
33             12.9  135675
34             13.5  139465>
```

In [20]:

```
df.isnull().sum()
```

Out[20]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [21]:

```
df.head()
```

Out[21]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [22]:

```
df.columns
```

Out[22]:

```
Index(['YearsExperience', 'Salary'], dtype='object')
```

In [23]:

```
df.loc[4,"Salary"]
```

Out[23]:

```
39891
```

In [24]:

```
df.head(15)
```

Out[24]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4.0	55794
12	4.0	56957
13	4.1	57081
14	4.5	61111

In [25]:

```
df.loc[2,"YearsExperience"]
```

Out[25]:

```
1.5
```

In [26]:

```
df.loc[12]
```

Out[26]:

```
YearsExperience        4.0
Salary             56957.0
Name: 12, dtype: float64
```

In [27]:

```
df.loc[4]
```

Out[27]:

```
YearsExperience        2.2
Salary             39891.0
Name: 4, dtype: float64
```

In [28]:

```
a=(1,2,3,4,5,6,7,8,9,10)
```

In [29]:

```
a[1:4]
```

Out[29]:

```
(2, 3, 4)
```

In [30]:

```
df.loc[0:3,'YearsExperience':"Salary"]
```

Out[30]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525

In [31]:

```
df.iloc[1,0]
```

Out[31]:

```
1.3
```

In [32]:

```
df.head()
```

Out[32]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [33]:

```
df.loc[1,"Salary"]
```

Out[33]:

```
46205
```

In [34]:

```
#Assigning values in X & Y
x=df.iloc[:, :-1].values
y=df.iloc[:, :-1].values
```

In [35]:
```
a[:2]
```
Out[35]:
```
(1, 2)
```

a[1:

In [36]:
```
a[2:]
```
Out[36]:
```
(3, 4, 5, 6, 7, 8, 9, 10)
```

In [37]:
```
a[1:6:2]
```
Out[37]:
```
(2, 4, 6)
```

In [38]:
```
a[1:6:2]
```
Out[38]:
```
(2, 4, 6)
```

In [39]:
```
print(x)
```
```
[[ 1.1]
 [ 1.3]
 [ 1.5]
 [ 2. ]
 [ 2.2]
 [ 2.9]
 [ 3. ]
 [ 3.2]
 [ 3.2]
 [ 3.7]
 [ 3.9]
 [ 4. ]
 [ 4. ]
 [ 4.1]
 [ 4.5]
 [ 4.9]
 [ 5.1]
 [ 5.3]
 [ 5.9]
 [ 6. ]
 [ 6.8]
 [ 7.1]
 [ 7.9]
 [ 8.2]
 [ 8.7]
 [ 9. ]
 [ 9.5]
 [ 9.6]
 [10.3]
 [10.5]
 [11.2]
 [11.5]
 [12.3]
 [12.9]
 [13.5]]
```

In [40]:
```
print(y)
```

```
[[ 1.1]
 [ 1.3]
 [ 1.5]
 [ 2. ]
 [ 2.2]
 [ 2.9]
 [ 3. ]
 [ 3.2]
 [ 3.2]
 [ 3.7]
 [ 3.9]
 [ 4. ]
 [ 4. ]
 [ 4.1]
 [ 4.5]
 [ 4.9]
 [ 5.1]
 [ 5.3]
 [ 5.9]
 [ 6. ]
 [ 6.8]
 [ 7.1]
 [ 7.9]
 [ 8.2]
 [ 8.7]
 [ 9. ]
 [ 9.5]
 [ 9.6]
 [10.3]
 [10.5]
 [11.2]
 [11.5]
 [12.3]
 [12.9]
 [13.5]]
```

In [41]:

```
#splitting testdata into x_tarin,y_train'
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.3, random_state=42)
```

In [42]:

```
print(x_train)
```

```
[[12.9]
 [ 1.1]
 [ 2.2]
 [ 5.3]
 [ 9.6]
 [ 2.9]
 [ 4. ]
 [ 1.3]
 [ 1.5]
 [12.3]
 [ 2. ]
 [11.2]
 [ 8.2]
 [11.5]
 [ 3.9]
 [ 7.9]
 [ 5.9]
 [ 9. ]
 [ 3. ]
 [ 6.8]
 [13.5]
 [ 3.2]
 [ 4.5]
 [10.3]]
```

In [43]:

```
print(x_test)
```

```
[[ 9.5]
 [ 4.1]
 [ 8.7]
 [ 7.1]
 [ 4.9]
 [10.5]
 [ 6. ]
 [ 4. ]
 [ 3.2]
 [ 5.1]
 [ 3.7]]
```

In [44]:

```
print(y_train)
```

```
[[12.9]
 [ 1.1]
 [ 2.2]
 [ 5.3]
 [ 9.6]
 [ 2.9]
 [ 4. ]
 [ 1.3]
 [ 1.5]
 [12.3]
 [ 2. ]
 [11.2]
 [ 8.2]
 [11.5]
 [ 3.9]
 [ 7.9]
 [ 5.9]
 [ 9. ]
 [ 3. ]
 [ 6.8]
 [13.5]
 [ 3.2]
 [ 4.5]
 [10.3]]
```

In [45]:

```
print(y_test)
```

```
[[ 9.5]
 [ 4.1]
 [ 8.7]
 [ 7.1]
 [ 4.9]
 [10.5]
 [ 6. ]
 [ 4. ]
 [ 3.2]
 [ 5.1]
 [ 3.7]]
```

In [46]:

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[46]:

```
▾ LinearRegression
LinearRegression()
```

In [47]:

```
#Assigning coefficient (slope) to m
m = lr.coef_
```

In [48]:

```
print("Coefficient :",a)
```

```
Coefficient : (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

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