Importing Libraries

In [1]:

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

Load Dataset from Device

In [2]:

```
from google.colab import files
uploaded = files.upload()
```

Choose File **No file selected**

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving data.csv to data.csv
```

Load Dataset

In [8]:

```
dataset = pd.read_csv('data.csv')
```

In [9]:

```
print(dataset.shape)
print(dataset.head(5))
```

```
(94, 2)
             x           y
0  168.181818  160.840244
1  187.878788  159.413657
2  207.575758  157.136809
3  227.272727  159.357847
4  246.969697  157.542862
```

Segregate Dataset into Input - X & Output - Y

In [10]:

```
X = dataset.iloc[:, :-1].values
X
```

Out[10]:

```
array([[ 168.18181818],
       [ 187.87878788],
       [ 207.57575758],
       [ 227.27272727],
       [ 246.96969697],
       [ 266.66666667],
       [ 286.36363636],
       [ 306.06060606],
       [ 325.75757576],
       [ 345.45454545],
       [ 365.15151515],
       [ 384.84848485],
       [ 404.54545455],
       [ 424.24242424],
       [ 443.93939394],
       [ 463.63636364],
       [ 483.33333333],
```

```
[ 503.03030303],
[ 522.72727273],
[ 542.42424242],
[ 562.12121212],
[ 581.81818182],
[ 601.51515152],
[ 621.21212121],
[ 640.90909091],
[ 660.60606061],
[ 680.3030303 ],
[ 700.        ],
[ 719.6969697 ],
[ 739.39393939],
[ 759.09090909],
[ 778.78787879],
[ 798.48484848],
[ 818.18181818],
[ 837.87878788],
[ 857.57575758],
[ 877.27272727],
[ 896.96969697],
[ 916.66666667],
[ 936.36363636],
[ 956.06060606],
[ 975.75757576],
[ 995.45454545],
[1015.15151515],
[1034.84848485],
[1054.54545455],
[1074.24242424],
[1093.93939394],
[1113.63636364],
[1133.33333333],
[1153.03030303],
[1172.72727273],
[1192.42424242],
[1212.12121212],
[1231.81818182],
[1251.51515152],
[1271.21212121],
[1290.90909091],
[1310.60606061],
[1330.3030303 ],
[1350.        ],
[1369.6969697 ],
[1389.39393939],
[1409.09090909],
[1428.78787879],
[1448.48484848],
[1468.18181818],
[1487.87878788],
[1507.57575758],
[1527.27272727],
[1546.96969697],
[1566.66666667],
[1586.36363636],
[1606.06060606],
[1625.75757576],
[1645.45454545],
[1665.15151515],
[1684.84848485],
[1704.54545455],
[1724.24242424],
[1743.93939394],
[1763.63636364],
[1783.33333333],
[1803.03030303],
[1822.72727273],
[1842.42424242],
[1862.12121212],
[1881.81818182],
[1901.51515152],
```

```
       [1921.21212121],
       [1940.90909091],
       [1960.60606061],
       [1980.3030303 ],
       [2000.        ]])
```

In [11]:

```
Y = dataset.iloc[:, -1].values
Y
```

Out[11]:

```
array([160.84024381, 159.41365734, 157.1368088 , 159.35784736,
       157.54286158, 157.73520716, 159.34756091, 155.23404557,
       155.80774009, 158.3299704 , 157.62585291, 160.47697951,
       158.22940639, 157.41781684, 163.37069148, 160.18481104,
       160.96838974, 158.18080666, 160.13850728, 161.6460876 ,
       159.31922497, 162.56957785, 160.81387414, 161.62873371,
       161.20567768, 166.31061698, 162.77603585, 160.88457814,
       164.84205952, 160.95225209, 164.00863628, 159.86853854,
       161.32847639, 164.57554065, 165.85572104, 164.91849414,
       164.54143071, 164.36748958, 162.20962269, 163.92394795,
       164.63932852, 167.87182021, 166.64178203, 162.62543484,
       166.99665279, 165.77528998, 165.38858024, 168.16274652,
       169.19836268, 169.19589357, 165.85186798, 167.10884798,
       168.58676929, 170.07230238, 167.35983334, 168.14383356,
       166.49945126, 166.51667766, 170.73111225, 172.01551036,
       169.35597976, 171.70403549, 170.61721144, 168.80066958,
       171.01067   , 173.56092162, 170.6101661 , 174.00807519,
       165.83626737, 172.91653228, 171.64379111, 171.06865197,
       172.04715792, 168.08546823, 171.81823198, 173.1687706 ,
       175.60730324, 171.81194441, 171.42846734, 172.23891016,
       175.27019817, 174.29386586, 172.77381293, 175.0568379 ,
       174.42142783, 176.36153241, 173.21710593, 174.16285752,
       174.23093521, 172.28509132, 176.00133146, 176.12817115,
       175.81325722, 175.53082573])
```

Splitting Dataset into Testing our Model

In [13]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size = 0.20, random_state
= 0)
```

Training Dataset using Support Vector Regression

In [15]:

```
from sklearn.svm import SVR
model = SVR()
model.fit(x_train, y_train)
```

Out[15]:

```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

Prediction for all Test data

In [20]:

```
ypred = model.predict(x_test)

from sklearn.metrics import r2_score, mean_squared_error
mse = mean_squared_error(y_test, ypred)
rmse = np.sqrt(mse)
print("Root Mean Square Error : ",rmse)
r2score = r2_score(y_test, ypred)
```

```
print("R2Score", r2score*100)
```

```
Root Mean Square Error :  2.35947188444521
R2Score 86.64242653738361
```