Machine Learning & Deep Learning

635 | Vishal Baraiya | 23010101014

Lab - 5

SVR

Importing the libraries

In [1]:

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

Read World bank CSV

In [2]:

```
df = pd.read_csv("WorldBank.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	
0	India	IND	Export value index (2000 = 100)	TX.VAL.MRCH.XD.WD	NaN	NaN	NaN	
1	India	IND	Insurance and financial services (% of commerc...	TX.VAL.INSF.ZS.WT	NaN	NaN	NaN	
2	India	IND	Merchandise imports by the reporting economy, ...	TM.VAL.MRCH.RS.ZS	4.983551	6.48805	10.124611	9.45
3	India	IND	Food imports (% of merchandise imports)	TM.VAL.FOOD.ZS.UN	NaN	NaN	17.080013	15.19
4	India	IND	Share of tariff lines with international peaks...	TM.TAX.MRCH.IP.ZS	NaN	NaN	NaN	

5 rows × 65 columns

Perform conditional selection to find - Population ages 15-64 (% of total population)

In [21]:
```
df1 = df[df['Indicator Name'] == 'Population ages 15-64 (% of total population)']
```

In [22]:
```
df1
```

Out[22]:

	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1
9	India	IND	Population ages 15-64 (% of total population)	SP.POP.1564.TO.ZS	56.49748	56.177532	55.807455	55.461

1 rows × 65 columns

Divide the data into input and output

```
In [23]: X = np.arange(1960,2020).reshape(-1,1)
```

```
In [40]: y = df1.values[0][4:-1:].reshape(-1,1)
         # y = df1.values[0][4:-1:]
```

```
In [41]: y
```

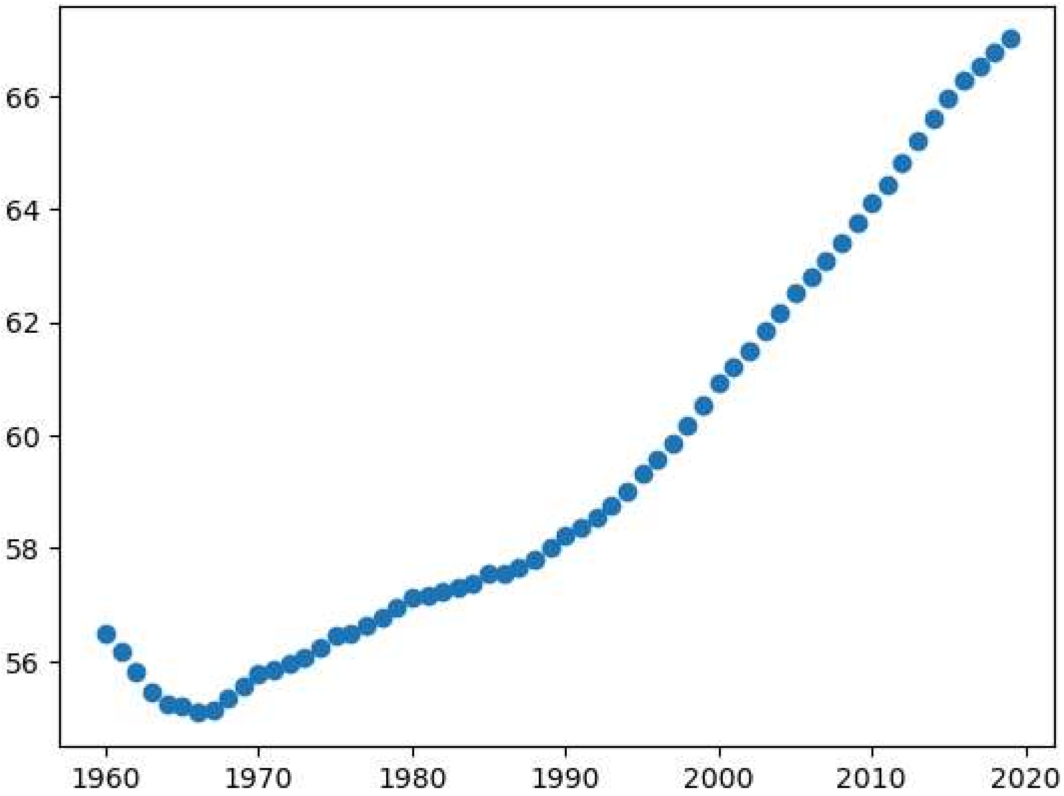
```
Out[41]:  array([[56.49748004],
                 [56.17753236],
                 [55.80745463],
                 [55.46166361],
                 [55.24893881],
                 [55.21135053],
                 [55.09090078],
                 [55.15534672],
                 [55.34507283],
                 [55.57014408],
                 [55.78194745],
                 [55.85676846],
                 [55.95268174],
                 [56.07247186],
                 [56.23447551],
                 [56.44405309],
                 [56.49722595],
                 [56.62068516],
                 [56.78900152],
                 [56.9691436],
                 [57.1425581],
                 [57.18105454],
                 [57.22630775],
                 [57.28875038],
                 [57.39054366],
                 [57.54142108],
                 [57.55350984],
                 [57.6545119],
                 [57.81875323],
                 [58.01501187],
                 [58.22990246],
                 [58.37403848],
                 [58.5472698],
                 [58.75605047],
                 [59.012126],
                 [59.31657719],
                 [59.56507329],
                 [59.8572303],
                 [60.18600058],
                 [60.53971518],
                 [60.90862046],
                 [61.18898716],
                 [61.4993847],
                 [61.83084479],
                 [62.173897],
                 [62.52276485],
                 [62.80842981],
                 [63.10261029],
                 [63.40924784],
                 [63.74196691],
                 [64.10821053],
                 [64.429404],
                 [64.80551944],
                 [65.20848906],
                 [65.5959799],
                 [65.94416405],
       [66.27426247],
       [66.53818711],
       [66.7667425],
       [67.00381119]], dtype=object)
```

Plot scatter plot of Population ages 15-64 (% of total population)

In [42]:
```
import matplotlib.pyplot as plt
```

In [44]:
```
plt.scatter(X,y)
```

Out[44]:
```
<matplotlib.collections.PathCollection at 0x2359d1b6f60>
```



Feature Scaling (Mandatory for SVR)**

SVR is highly sensitive to the range of data points. If we don't scale (normalize) the data, the model will fail to find the correct hyperplane.

In [45]:
```
from sklearn.preprocessing import StandardScaler
```

In [47]:
```
scaler_x = StandardScaler()
scaler_y = StandardScaler()

x_scaled = scaler_x.fit_transform(X)
```

```
y_scaled = scaler_y.fit_transform(y)

print("Scaling Complete")
```

```
Scaling Complete
```

In [ ]:

Splitting the dataset into the Training set and Test set

In [ ]:

In [ ]:

In [ ]:

Fitting SVR on 3 Different Kernel on dataset

In [65]:

```
from sklearn.svm import SVR
```

In [66]:

```
svr_linear = SVR()
```

In [67]:

```
svr_linear.fit(x_scaled,y_scaled)
```

```
D:\Anaconda\Lib\site-packages\sklearn\utils\validation.py:1339: DataConversionWarnin
g: A column-vector y was passed when a 1d array was expected. Please change the shap
e of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[67]:

```
▾ SVR
SVR()
```

In [68]:

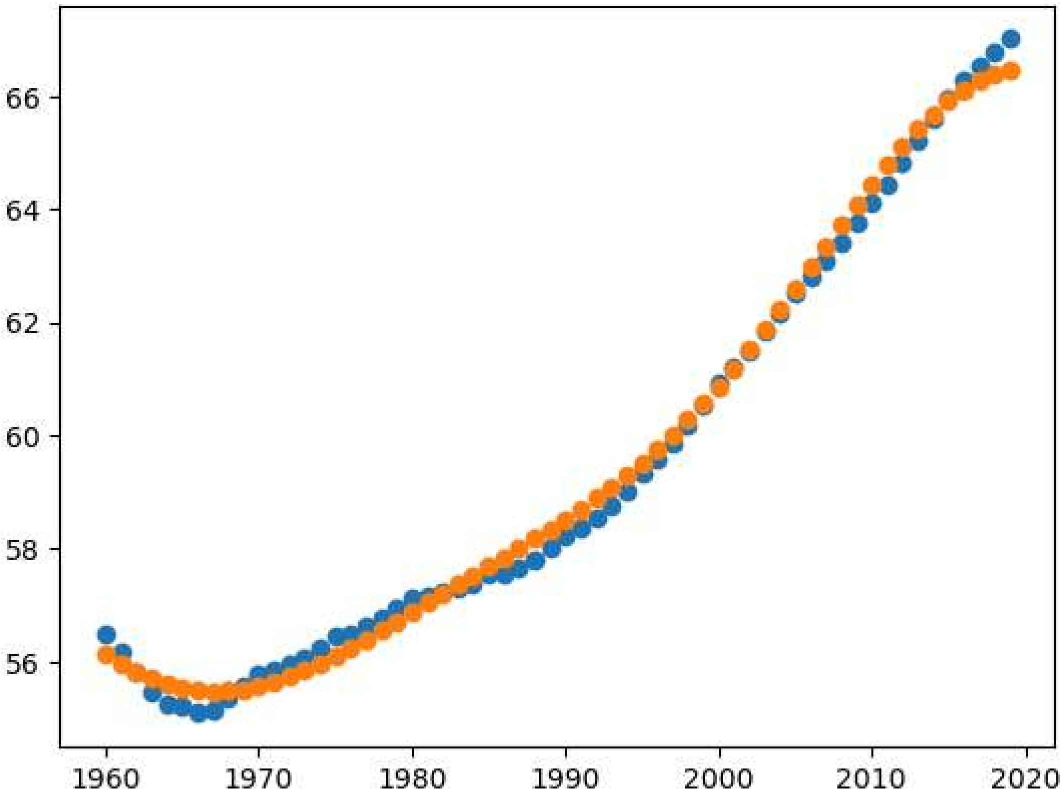
```
y_predited = scaler_y.inverse_transform(svr_linear.predict(x_scaled).reshape(-1,1))
```

In [69]:

```
plt.scatter(X,y)
plt.scatter(X,y_predited)
```

Out[69]:

```
<matplotlib.collections.PathCollection at 0x2359db66060>
```



```
In [ ]:
```

```
In [61]:
```

Predict the x_test using 3 Kernel

```
In [ ]: model_rbf.score(x_train,y_train)

Out[ ]: 0.9947138713539011
```

```
In [ ]: model_rbf.score(x_test,y_test)

Out[ ]: 0.9880623181593732
```

```
In [ ]:
```

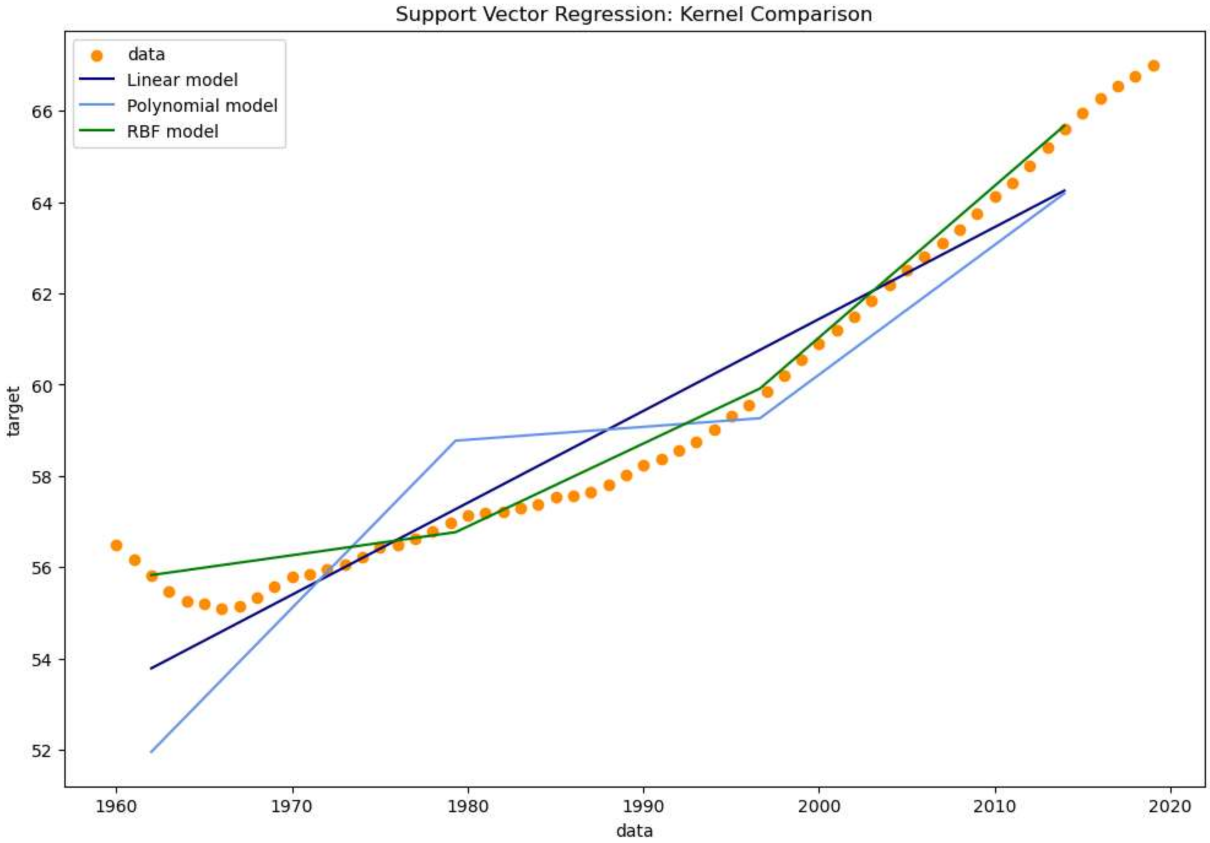
```
In [ ]:

Out[ ]: array([57.55350984, 59.31657719, 67.00381119, 57.81875323, 55.85676846,
               55.80745463, 59.012126, 66.7667425, 60.90862046, 57.22630775,
               55.24893881, 55.78194745, 58.22990246, 61.18898716, 58.75605047,
               61.83084479, 63.74196691, 55.15534672], dtype=object)
```

```
In [ ]:
```

```
Out[ ]: array([57.73445326, 59.30206118, 64.8451716 , 57.99812952, 55.96479236,
               56.12587167, 59.06485884, 64.91876466, 60.76906052, 57.24445561,
               55.93360816, 55.89821768, 58.29641078, 61.1088798 , 58.84682535,
               61.81299688, 63.81278461, 55.81527994])
```

Visualising the results

```
In [ ]:
```



Student Activity : Prediction

Task: Predict the value for the Year **2025** (or value 6.5 in the demo data). Remember: You must transform the input before predicting, and inverse transform the output.

```
In [75]: newData = [[2025]]
```

```
In [76]: predicted = svr_linear.predict(scaler_x.transform(newData))
```

```
In [77]: scaler_y.inverse_transform(predicted.reshape(-1,1))
```

```
Out[77]: array([[65.91120533]])
```

```
In [ ]:
```