PROJECT 3

NAME- ANURAG MISHRA SIC- 20BCED17

Data Preprocessing of the dataset 'investment_data'

Step1 : Importing the libraries

In [61]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Step 2: Import data set

In [62]:

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

In [63]:

```
dataset
```

Out[63]:

	year	per capita income (US$)
0	1970	3399.299037
1	1971	3768.297935
2	1972	4251.175484
3	1973	4804.463248
4	1974	5576.514583
5	1975	5998.144346
6	1976	7062.131392
7	1977	7100.126170
8	1978	7247.967035
9	1979	7602.912681
10	1980	8355.968120
11	1981	9434.390652
12	1982	9619.438377
13	1983	10416.536590
14	1984	10790.328720
15	1985	11018.955850
16	1986	11482.891530
17	1987	12974.806620
18	1988	15080.283450
19	1989	16426.725480
20	1990	16838.673200

	year	per capita income (US$)
21	1991	17266.097690
22	1992	16412.083090
23	1993	15875.586730
24	1994	15755.820270
25	1995	16369.317250
26	1996	16699.826680
27	1997	17310.757750
28	1998	16622.671870
29	1999	17581.024140
30	2000	18987.382410
31	2001	18601.397240
32	2002	19232.175560
33	2003	22739.426280
34	2004	25719.147150
35	2005	29198.055690
36	2006	32738.262900
37	2007	36144.481220
38	2008	37446.486090
39	2009	32755.176820
40	2010	38420.522890
41	2011	42334.711210
42	2012	42665.255970
43	2013	42676.468370
44	2014	41039.893600
45	2015	35175.188980
46	2016	34229.193630

Step3: To create feature matrix and dependent variable vector

In [64]:

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

In [65]:

```
x
```

Out[65]:

```
array([[1970],
       [1971],
       [1972],
       [1973],
       [1974],
       [1975],
       [1976],
       [1977],
       [1978],
       [1979],
       [1980],
       [1981],
       [1982],
```

```
       [1983],
       [1984],
       [1985],
       [1986],
       [1987],
       [1988],
       [1989],
       [1990],
       [1991],
       [1992],
       [1993],
       [1994],
       [1995],
       [1996],
       [1997],
       [1998],
       [1999],
       [2000],
       [2001],
       [2002],
       [2003],
       [2004],
       [2005],
       [2006],
       [2007],
       [2008],
       [2009],
       [2010],
       [2011],
       [2012],
       [2013],
       [2014],
       [2015],
       [2016]], dtype=int64)
```

In [66]:

```
y
```

Out[66]:

```
array([ 3399.299037,  3768.297935,  4251.175484,  4804.463248,
        5576.514583,  5998.144346,  7062.131392,  7100.12617 ,
        7247.967035,  7602.912681,  8355.96812 ,  9434.390652,
        9619.438377, 10416.53659 , 10790.32872 , 11018.95585 ,
       11482.89153 , 12974.80662 , 15080.28345 , 16426.72548 ,
       16838.6732  , 17266.09769 , 16412.08309 , 15875.58673 ,
       15755.82027 , 16369.31725 , 16699.82668 , 17310.75775 ,
       16622.67187 , 17581.02414 , 18987.38241 , 18601.39724 ,
       19232.17556 , 22739.42628 , 25719.14715 , 29198.05569 ,
       32738.2629  , 36144.48122 , 37446.48609 , 32755.17682 ,
       38420.52289 , 42334.71121 , 42665.25597 , 42676.46837 ,
       41039.8936  , 35175.18898 , 34229.19363 ])
```

Step4: Replace missing data

In [67]:

```
from sklearn.impute import SimpleImputer
imputer=SimpleImputer(missing_values=np.nan,strategy='mean')
imputer.fit(x[:,:])
x[:,:]=imputer.transform(x[:,:])
```

In [68]:

```
x
```

Out[68]:

```
array([[1970],
       [1971],
```

```
       [1971],
       [1972],
       [1973],
       [1974],
       [1975],
       [1976],
       [1977],
       [1978],
       [1979],
       [1980],
       [1981],
       [1982],
       [1983],
       [1984],
       [1985],
       [1986],
       [1987],
       [1988],
       [1989],
       [1990],
       [1991],
       [1992],
       [1993],
       [1994],
       [1995],
       [1996],
       [1997],
       [1998],
       [1999],
       [2000],
       [2001],
       [2002],
       [2003],
       [2004],
       [2005],
       [2006],
       [2007],
       [2008],
       [2009],
       [2010],
       [2011],
       [2012],
       [2013],
       [2014],
       [2015],
       [2016]], dtype=int64)
```

Step5: Encoding

step6 : spiliting of data set into training and testing set

In [69]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=1)
```

In [70]:

```
xtrain
```

Out[70]:

```
array([[1989],
       [2006],
       [2016],
       [2003],
       [1993],
       [2004],
       [1997],
       [1991]
```

```
       [1983],
       [2008],
       [1987],
       [2012],
       [1974],
       [1998],
       [1984],
       [1980],
       [2011],
       [2000],
       [2001],
       [2010],
       [1990],
       [1988],
       [1995],
       [1976],
       [1977],
       [2014],
       [1971],
       [1986],
       [1970],
       [1985],
       [1975],
       [1981],
       [1979],
       [1978],
       [1982],
       [2013],
       [2007]], dtype=int64)
```

In [71]:

```
ytrain
```

Out[71]:

```
array([16426.72548 , 32738.2629  , 34229.19363 , 22739.42628 ,
       15875.58673 , 25719.14715 , 17310.75775 , 17266.09769 ,
       10416.53659 , 37446.48609 , 12974.80662 , 42665.25597 ,
        5576.514583, 16622.67187 , 10790.32872 ,  8355.96812 ,
       42334.71121 , 18987.38241 , 18601.39724 , 38420.52289 ,
       16838.6732  , 15080.28345 , 16369.31725 ,  7062.131392,
        7100.12617 , 41039.8936  ,  3768.297935, 11482.89153 ,
        3399.299037, 11018.95585 ,  5998.144346,  9434.390652,
        7602.912681,  7247.967035,  9619.438377, 42676.46837 ,
       36144.48122 ])
```

step7 : Feature scaling (not required)

Part B: build my first linear model

step 1: training the model

In [72]:

```
from sklearn.linear_model import LinearRegression
LR=LinearRegression()
LR.fit(xtrain,ytrain)
```

Out[72]:

```
LinearRegression()
```

step 2: testing the linear model

In [73]:

```
yestimated=LR.predict(xtest)
```

In [74]:

```
yestimated
```

Out[74]:

```
array([20349.94572643, 18613.49135581, 33373.35350612, 29900.44476487,
        1248.94764955,  2117.17483487, 24691.081653  , 27295.76320894,
       38582.716618  , 22086.40009706])
```

In [75]:

```
ytest
```

Out[75]:

```
array([15755.82027 , 16412.08309 , 32755.17682 , 29198.05569 ,
        4251.175484,  4804.463248, 17581.02414 , 19232.17556 ,
       35175.18898 , 16699.82668 ])
```

step 3: visualising the data's

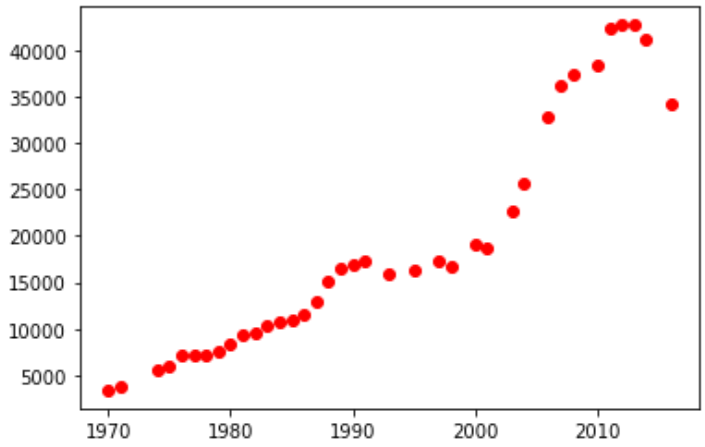
step a: training data

In [76]:

```
plt.scatter(xtrain,ytrain,color='red')
```

Out[76]:

```
<matplotlib.collections.PathCollection at 0x1bdf3633580>
```

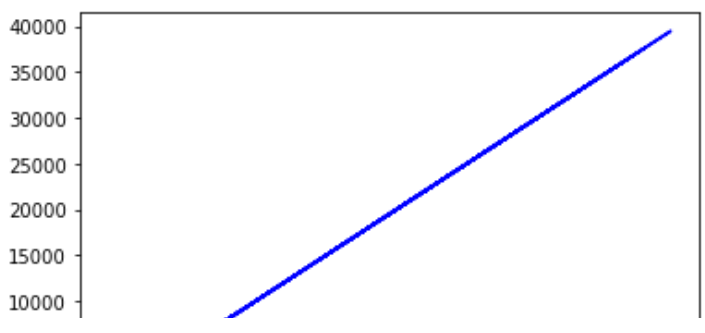


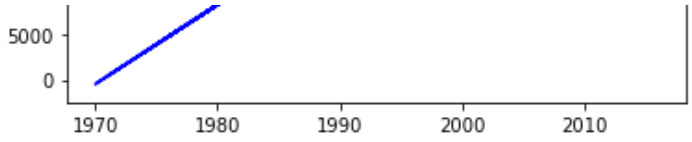
In [77]:

```
plt.plot(xtrain,LR.predict(xtrain),color='blue')
```

Out[77]:

```
[<matplotlib.lines.Line2D at 0x1bdf368e4c0>]
```



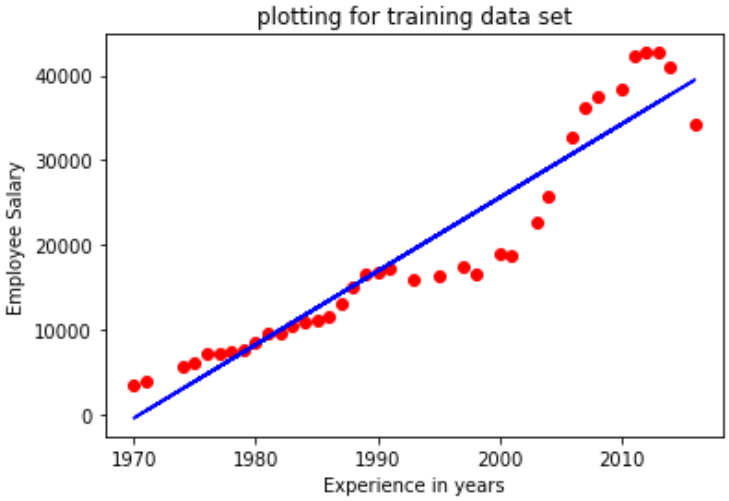


In [78]:

```
plt.scatter(xtrain,ytrain,color='red')
plt.plot(xtrain,LR.predict(xtrain),color='blue')
plt.xlabel('Experience in years')
plt.ylabel('Employee Salary')
plt.title('plotting for training data set')
```

Out[78]:

```
Text(0.5, 1.0, 'plotting for training data set')
```



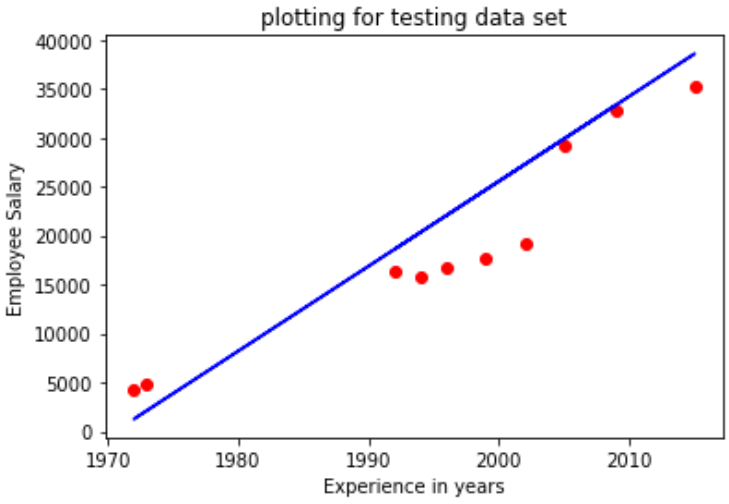
step b: testing data

In [79]:

```
plt.scatter(xtest,ytest,color='red')
plt.plot(xtest,LR.predict(xtest),color='blue')
plt.xlabel('Experience in years')
plt.ylabel('Employee Salary')
plt.title('plotting for testing data set')
```

Out[79]:

```
Text(0.5, 1.0, 'plotting for testing data set')
```



In [80]:

```
LR.coef_
```

Out[80]:

```
array([868.22718531])
```

In [83]:

```
LR.intercept_
```

Out[83]:

```
-1710895.0617872213
```

For the given predict the per capitia income for Canadian citizens in year 2021.

In [82]:

```
LR.predict([[5]])
```

Out[82]:

```
array([-1706553.92586066])
```

In [ ]:

```

```