--------------------------- Created By Vinay Kumar ---------------------------

Simple Linear Regression

```
Single Features
1 Feature vs 1 Target:Ex(Study hrs Vs Marks)
```

Multiple Liner Regression

```
Multiple Features
More than 1 Col(Features) vs 1 Target: Ex(Sales,Profit,Loan
Amount,GDP)
```

Score High(Best),Error(Lowest):Best Model

```
1:Error's: Loss, Residuals, Cost: Low>>
```

Canadas's Per Capita Income($)

Step 1 Load important Modules

In [35]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error,mean_squared_error,root_mean_square
import warnings
warnings.filterwarnings('ignore')
print('Done')
```

```
Done
```

Step2: Load Dataset

In [2]:

```
 file_path = r"C:\Users\Administrator\Downloads\canada_per_capita_income.csv"
df =pd.read_csv(file_path)
print('Done')
```

```
Done
```

EDA:-Exploratory Data Analysis

In [3]:
```
df.shape
```

Out[3]:
```
(47, 2)
```

In [4]:
```
df.sample(5)
```

Out[4]:

	year	per capita income (US$)
19	1989	16426.725480
37	2007	36144.481220
7	1977	7100.126170
34	2004	25719.147150
11	1981	9434.390652

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 2 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   year                     47 non-null     int64  
 1   per capita income (US$)  47 non-null     float64
dtypes: float64(1), int64(1)
memory usage: 884.0 bytes
```

In [6]:
```
df.head()
```

Out[6]:

	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

In [7]:
```
df.isna().sum()
```

Out[7]:
```
year                       0
per capita income (US$)    0
dtype: int64
```

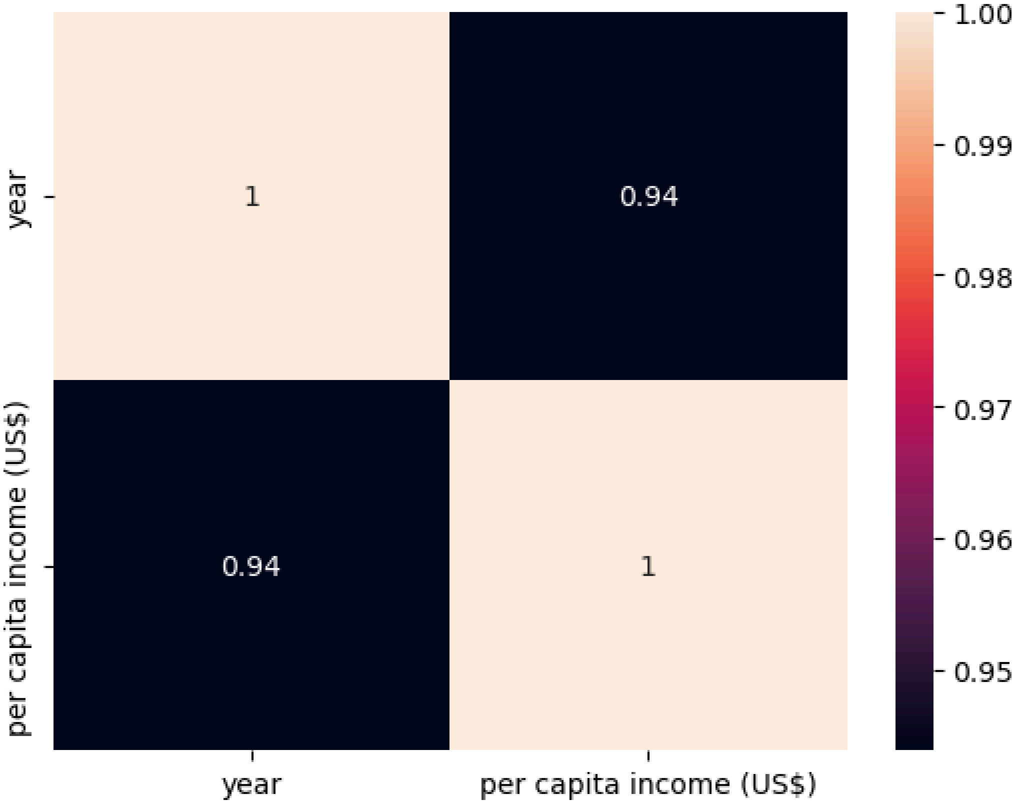
In [8]:
```
df.describe()
```

Out[8]:

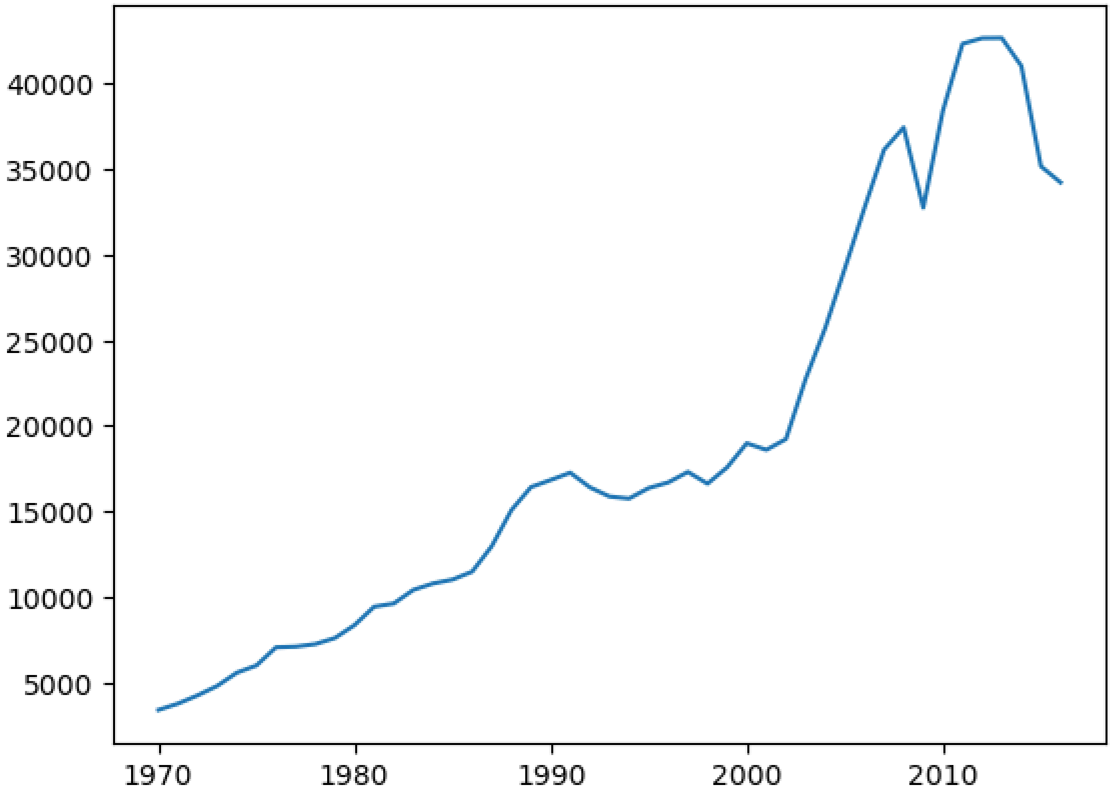
	year	per capita income (US$)
count	47.000000	47.000000
mean	1993.000000	18920.137063
std	13.711309	12034.679438
min	1970.000000	3399.299037
25%	1981.500000	9526.914515
50%	1993.000000	16426.725480
75%	2004.500000	27458.601420
max	2016.000000	42676.468370

In [9]:
```
corr= df.corr()
```

In [10]:
```
sns.heatmap(corr,annot=True)
plt.show()
```



In [11]:
```
plt.plot(df['year'],df['per capita income (US$)'])
plt.show()
```



Step3 Dataset Divide X,y

In [12]:
```
#3.1
X = df[['year']]
y = df['per capita income (US$)']

X.shape,y.shape
```

Out[12]:
```
((47, 1), (47,))
```

In [13]:
```
from sklearn.base import RegressorMixin
#3.2

model = LinearRegression()
model.fit(X,y)
```

Out[13]:
```
▾ LinearRegression ⓘ ?
LinearRegression()
```

In [14]:
```
y_pred = model.predict(X)
```

In [15]:
```
y_pred
```

```
Out[15]: array([ -134.55966672,   693.9054085 ,  1522.37048373,  2350.83555895,
                 3179.30063417,  4007.7657094 ,  4836.23078462,  5664.69585984,
                 6493.16093506,  7321.62601029,  8150.09108551,  8978.55616073,
                 9807.02123595, 10635.48631118, 11463.9513864 , 12292.41646162,
                13120.88153685, 13949.34661207, 14777.81168729, 15606.27676251,
                16434.74183774, 17263.20691296, 18091.67198818, 18920.1370634 ,
                19748.60213863, 20577.06721385, 21405.53228907, 22233.9973643 ,
                23062.46243952, 23890.92751474, 24719.39258996, 25547.85766519,
                26376.32274041, 27204.78781563, 28033.25289085, 28861.71796608,
                29690.1830413 , 30518.64811652, 31347.11319175, 32175.57826697,
                33004.04334219, 33832.50841741, 34660.97349264, 35489.43856786,
                36317.90364308, 37146.3687183 , 37974.83379353])
```

```
In [ ]: # --------------------------------  Created By Vinay Kumar   ----------------------
```

```
In [29]: compare_df =df.copy()
```

```
In [32]: compare_df['Y-Pred'] = y_pred
```

```
In [33]: compare_df.columns = ['X','y','y-pred']
```

```
In [34]: compare_df
```

Out[34]:

	X	y	y-pred
0	1970	3399.299037	-134.559667
1	1971	3768.297935	693.905409
2	1972	4251.175484	1522.370484
3	1973	4804.463248	2350.835559
4	1974	5576.514583	3179.300634
5	1975	5998.144346	4007.765709
6	1976	7062.131392	4836.230785
7	1977	7100.126170	5664.695860
8	1978	7247.967035	6493.160935
9	1979	7602.912681	7321.626010
10	1980	8355.968120	8150.091086
11	1981	9434.390652	8978.556161
12	1982	9619.438377	9807.021236
13	1983	10416.536590	10635.486311
14	1984	10790.328720	11463.951386
15	1985	11018.955850	12292.416462
16	1986	11482.891530	13120.881537
17	1987	12974.806620	13949.346612
18	1988	15080.283450	14777.811687
19	1989	16426.725480	15606.276763
20	1990	16838.673200	16434.741838
21	1991	17266.097690	17263.206913
22	1992	16412.083090	18091.671988
23	1993	15875.586730	18920.137063
24	1994	15755.820270	19748.602139
25	1995	16369.317250	20577.067214
26	1996	16699.826680	21405.532289
27	1997	17310.757750	22233.997364
28	1998	16622.671870	23062.462440
29	1999	17581.024140	23890.927515

	X	y	y-pred
30	2000	18987.382410	24719.392590
31	2001	18601.397240	25547.857665
32	2002	19232.175560	26376.322740
33	2003	22739.426280	27204.787816
34	2004	25719.147150	28033.252891
35	2005	29198.055690	28861.717966
36	2006	32738.262900	29690.183041
37	2007	36144.481220	30518.648117
38	2008	37446.486090	31347.113192
39	2009	32755.176820	32175.578267
40	2010	38420.522890	33004.043342
41	2011	42334.711210	33832.508417
42	2012	42665.255970	34660.973493
43	2013	42676.468370	35489.438568
44	2014	41039.893600	36317.903643
45	2015	35175.188980	37146.368718
46	2016	34229.193630	37974.833794

In [38]:

```
mae_math = (compare_df['y']  - compare_df['y-pred']).abs().mean()
print('MAE math',mae_math)
```

```
MAE math 3088.866427771443
```

In [39]:

```
mae = mean_absolute_error(compare_df['y'],compare_df['y-pred'])
print(mae)
```

```
3088.866427771443
```

In [41]:

```
mse_math = ((compare_df['y'] - compare_df['y-pred'])**2).mean()
print('MSE math',mse_math)
```

```
MSE math 15462739.061504772
```

In [42]:

```
mse = mean_squared_error(compare_df['y'],compare_df['y-pred'])
print(mse)
```

```
15462739.061504772
```

In [43]:

```
rmse_math = mse**0.5
print('RMSE math',rmse_math)
```

```
RMSE math 3932.268945723928
```

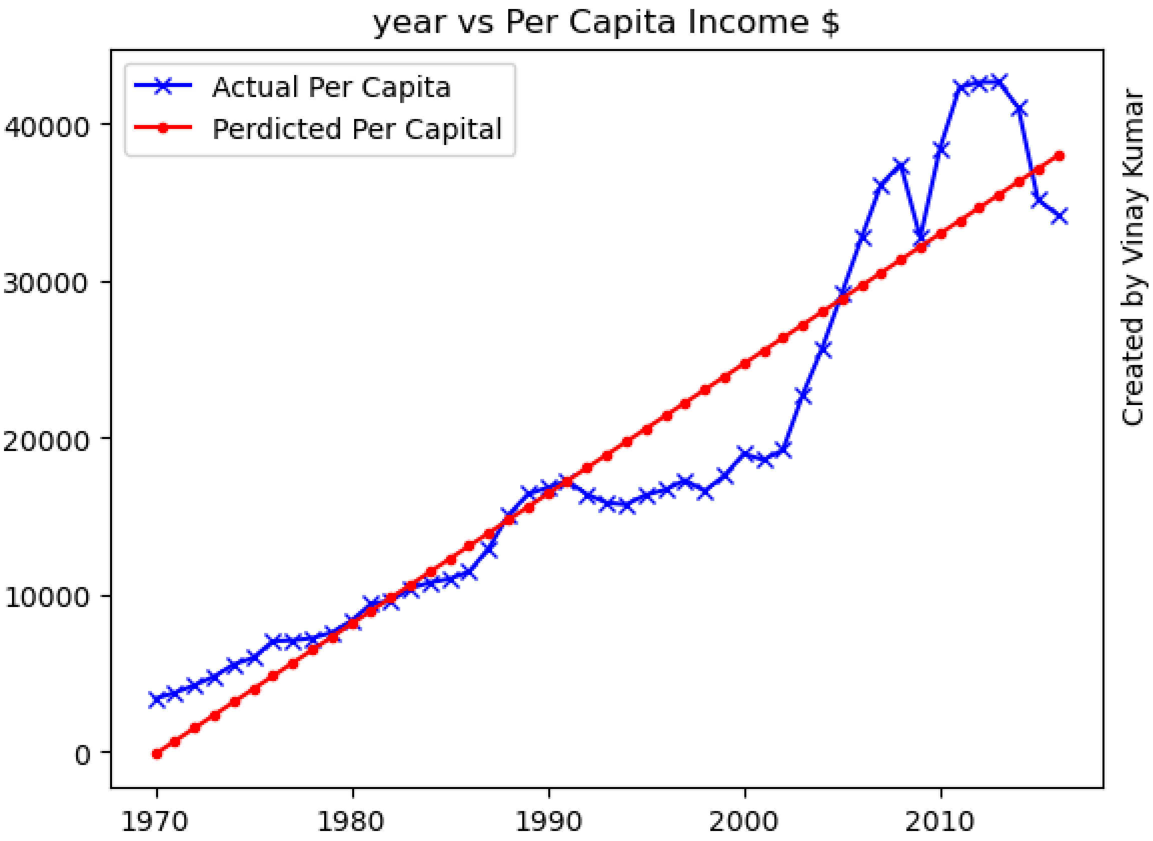
In [44]:
```
root_mean_squared_error(compare_df['y'],compare_df['y-pred'])
```

Out[44]:
```
3932.268945723928
```

https://cdn.analyticsvidhya.com/wp-content/uploads/2021/10/42439screenshot-2021-10-26-at-93408-pm-673f1d0b79f01.webp

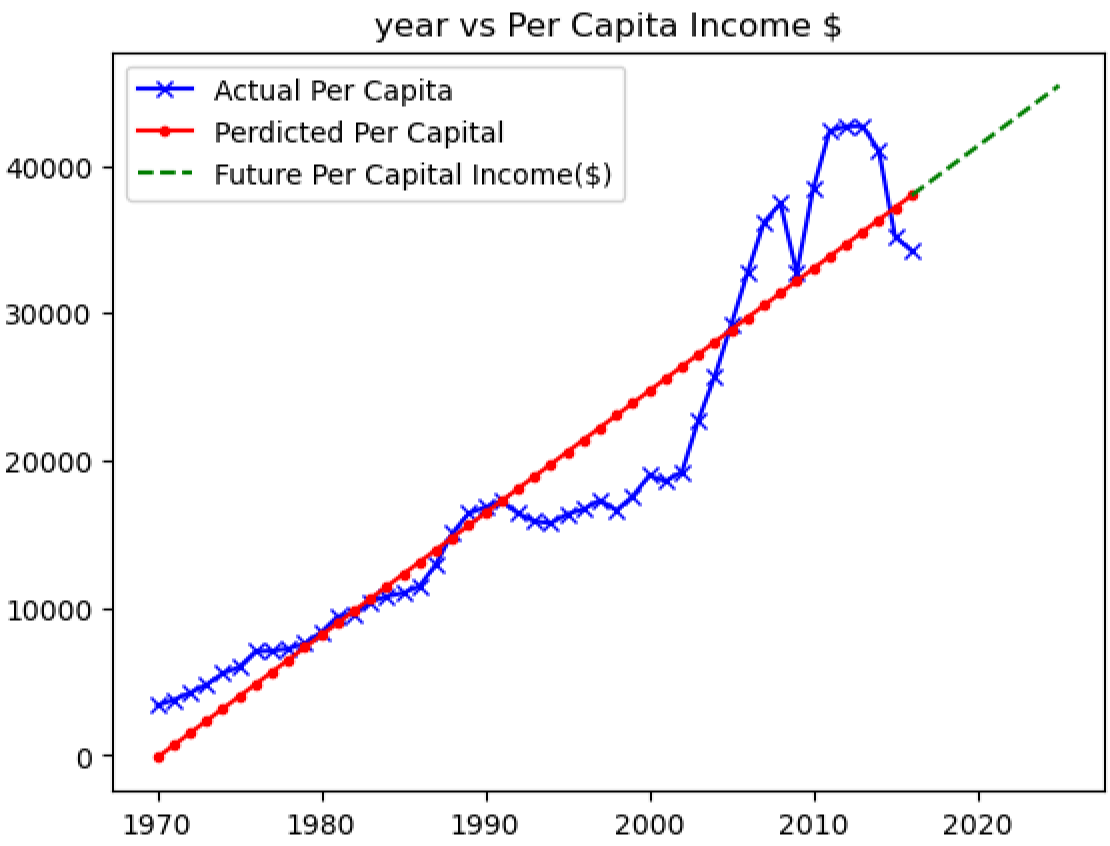
In [64]:
```
plt.title('year vs Per Capita Income $')
plt.plot(compare_df['X'],compare_df['y'],color = 'b',marker = 'x',label = 'Actual P
plt.plot(compare_df['X'],compare_df['y-pred'],color = 'r',label = 'Perdicted Per Ca
plt.legend()
# plt.annotate('Created by Vinay Kumar',
#              xy=(1.02, 0.5),
#              xycoords='axes fraction',
#              rotation=90,
#              fontsize=10)

plt.show()
```



In [65]:
```
next_years = np.arange(2016,2026).reshape((10,1))
next_years
future_per_capital = model.predict(next_years)
```

In [66]:
```
plt.title('year vs Per Capita Income $')
plt.plot(compare_df['X'],compare_df['y'],color = 'b',marker = 'x',label = 'Actual P
plt.plot(compare_df['X'],compare_df['y-pred'],color = 'r',label = 'Perdicted Per Ca
plt.plot(next_years,future_per_capital,color = 'g',label ='Future Per Capital Incom
plt.legend()
plt.show()
```



In [62]:

```
future_per_capital
```

Out[62]:

```
array([37974.83379353, 38803.29886875, 39631.76394397, 40460.22901919,
       41288.69409442, 42117.15916964, 42945.62424486, 43774.08932009,
       44602.55439531, 45431.01947053])
```

------------------------- Created By Vinay Kumar -------------------------

In [ ]: