Problem Statement

Linear Regression

Import Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
a=pd.read_csv("world.csv")
a
```

Out[2]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0
3	Andorra	164	AD	40.00%	468	NaN	7.20	376.0
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0
...	...	...	...	...	...	...	...	...
190	Venezuela	32	VE	24.50%	912,050	343,000	17.88	58.0
191	Vietnam	314	VN	39.30%	331,210	522,000	16.75	84.0
192	Yemen	56	YE	44.60%	527,968	40,000	30.45	967.0
193	Zambia	25	ZM	32.10%	752,618	16,000	36.19	260.0
194	Zimbabwe	38	ZW	41.90%	390,757	51,000	30.68	263.0

195 rows × 35 columns

To display top 10 rows

In [3]:
```
c=a.head(15)
c
```

Out[3]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	C
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0	
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0	
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0	
3	Andorra	164	AD	40.00%	468	NaN	7.20	376.0	
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	
5	Antigua and Barbuda	223	AG	20.50%	443	0	15.33	1.0	
6	Argentina	17	AR	54.30%	2,780,400	105,000	17.02	54.0	
7	Armenia	104	AM	58.90%	29,743	49,000	13.99	374.0	
8	Australia	3	AU	48.20%	7,741,220	58,000	12.60	61.0	
9	Austria	109	AT	32.40%	83,871	21,000	9.70	43.0	
10	Azerbaijan	123	AZ	57.70%	86,600	82,000	14.00	994.0	
11	The Bahamas	39	BS	1.40%	13,880	1,000	13.97	1.0	
12	Bahrain	2,239	BH	11.10%	765	19,000	13.99	973.0	
13	Bangladesh	1,265	BD	70.60%	148,460	221,000	18.18	880.0	
14	Barbados	668	BB	23.30%	430	1,000	10.65	1.0	

15 rows × 35 columns

To find Missing values

In [4]:
```
c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 35 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Country                                 15 non-null     object
 1   Density
(P/Km2)                          15 non-null     object
 2   Abbreviation                            15 non-null     object
 3   Agricultural Land( %)                   15 non-null     object
 4   Land Area(Km2)                          15 non-null     object
 5   Armed Forces size                       14 non-null     object
 6   Birth Rate                              15 non-null     float64
 7   Calling Code                            15 non-null     float64
 8   Capital/Major City                      15 non-null     object
 9   Co2-Emissions                           15 non-null     object
 10  CPI                                     14 non-null     object
 11  CPI Change (%)                          14 non-null     object
 12  Currency-Code                           14 non-null     object
 13  Fertility Rate                          15 non-null     float64
```

```
 14  Forested Area (%)                             15 non-null     object
 15  Gasoline Price                                15 non-null     object
 16  GDP                                           15 non-null     object
 17  Gross primary education enrollment (%)        15 non-null     object
 18  Gross tertiary education enrollment (%)       14 non-null     object
 19  Infant mortality                              15 non-null     float64
 20  Largest city                                  15 non-null     object
 21  Life expectancy                               14 non-null     float64
 22  Maternal mortality ratio                      14 non-null     float64
 23  Minimum wage                                  13 non-null     object
 24  Official language                             15 non-null     object
 25  Out of pocket health expenditure              15 non-null     object
 26  Physicians per thousand                       15 non-null     float64
 27  Population                                    15 non-null     object
 28  Population: Labor force participation (%)     13 non-null     object
 29  Tax revenue (%)                               14 non-null     object
 30  Total tax rate                                14 non-null     object
 31  Unemployment rate                             13 non-null     object
 32  Urban_population                              15 non-null     object
 33  Latitude                                      15 non-null     float64
 34  Longitude                                     15 non-null     float64
dtypes: float64(9), object(26)
memory usage: 4.2+ KB
```

To display summary of statistics

In [5]:
```
a.describe()
```

Out[5]:

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Lati
count	189.000000	194.000000	188.000000	189.000000	187.000000	181.000000	188.000000	194.00
mean	20.214974	360.546392	2.698138	21.332804	72.279679	160.392265	1.839840	19.09
std	9.945774	323.236419	1.282267	19.548058	7.483661	233.502024	1.684261	23.96
min	5.900000	1.000000	0.980000	1.400000	52.800000	2.000000	0.010000	-40.90
25%	11.300000	82.500000	1.705000	6.000000	67.000000	13.000000	0.332500	4.54
50%	17.950000	255.500000	2.245000	14.000000	73.200000	53.000000	1.460000	17.27
75%	28.750000	506.750000	3.597500	32.700000	77.500000	186.000000	2.935000	40.12
max	46.080000	1876.000000	6.910000	84.500000	85.400000	1150.000000	8.420000	64.96

To display column heading

In [6]:
```
a.columns
```

Out[6]:
```
Index(['Country', 'Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land( %)',
       'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Code',
       'Capital/Major City', 'Co2-Emissions', 'CPI', 'CPI Change (%)',
       'Currency-Code', 'Fertility Rate', 'Forested Area (%)',
       'Gasoline Price', 'GDP', 'Gross primary education enrollment (%)',
       'Gross tertiary education enrollment (%)', 'Infant mortality',
       'Largest city', 'Life expectancy', 'Maternal mortality ratio',
       'Minimum wage', 'Official language', 'Out of pocket health expenditure',
       'Physicians per thousand', 'Population',
```

```
       'Population: Labor force participation (%)', 'Tax revenue (%)',
       'Total tax rate', 'Unemployment rate', 'Urban_population', 'Latitude',
       'Longitude'],
      dtype='object')
```

Pairplot

In [7]:
```
s=a.dropna(axis=1)
s
```

Out[7]:

	Country	Density\n(P/Km2)
0	Afghanistan	60
1	Albania	105
2	Algeria	18
3	Andorra	164
4	Angola	26
...	...	...
190	Venezuela	32
191	Vietnam	314
192	Yemen	56
193	Zambia	25
194	Zimbabwe	38

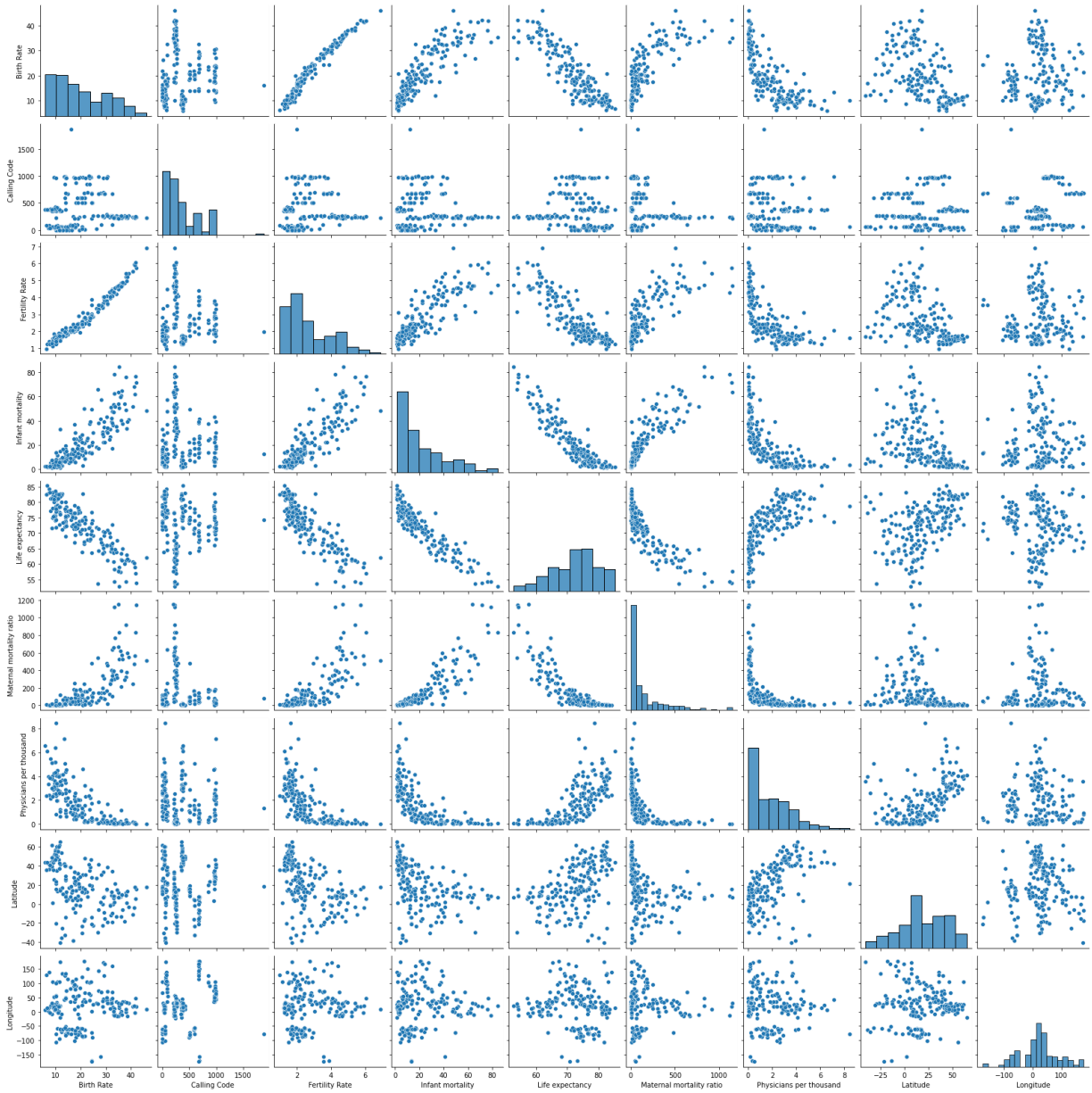
195 rows × 2 columns

In [8]:
```
s.columns
```

Out[8]:
```
Index(['Country', 'Density\n(P/Km2)'], dtype='object')
```

In [9]:
```
sns.pairplot(a)
```

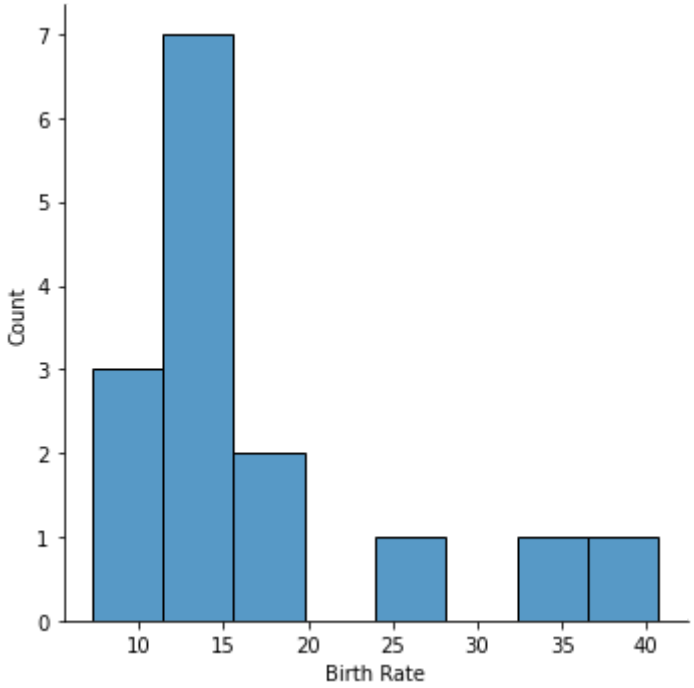
Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x2e32b0658b0>
```



Distribution Plot

In [16]:
```
sns.displot(c['Birth Rate'])
```

Out[16]:
```
<seaborn.axisgrid.FacetGrid at 0x2e32fe17a60>
```



Correlation

In [17]:
```
b=a[['Country', 'Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land( %)',
       'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Code',
       'Capital/Major City', 'Co2-Emissions', 'CPI', 'CPI Change (%)',
       'Currency-Code', 'Fertility Rate', 'Forested Area (%)',
       'Gasoline Price', 'GDP', 'Gross primary education enrollment (%)',
       'Gross tertiary education enrollment (%)', 'Infant mortality',
       'Largest city', 'Life expectancy', 'Maternal mortality ratio',
       'Minimum wage', 'Official language', 'Out of pocket health expenditure',
       'Physicians per thousand', 'Population',
       'Population: Labor force participation (%)', 'Tax revenue (%)',
       'Total tax rate', 'Unemployment rate', 'Urban_population', 'Latitude',
       'Longitude']]
sns.heatmap(b.corr())
```

Out[17]: <AxesSubplot:>

Train the model - Model Building

In [18]:
```
g=c[['Birth Rate']]
h=c['Birth Rate']
```

To split dataset into training end test

In [19]:
```
from sklearn.model_selection import train_test_split
g_train,g_test,h_train,h_test=train_test_split(g,h,test_size=0.6)
```

To run the model

In [20]:
```
from sklearn.linear_model import LinearRegression
```

In [21]:
```
lr=LinearRegression()
lr.fit(g_train,h_train)
```

Out[21]:
```
LinearRegression()
```

In [22]:
```
print(lr.intercept_)
```

```
0.0
```

Coeffecient

In [23]:
```
coeff=pd.DataFrame(lr.coef_,g.columns,columns=['Co-effecient'])
coeff
```

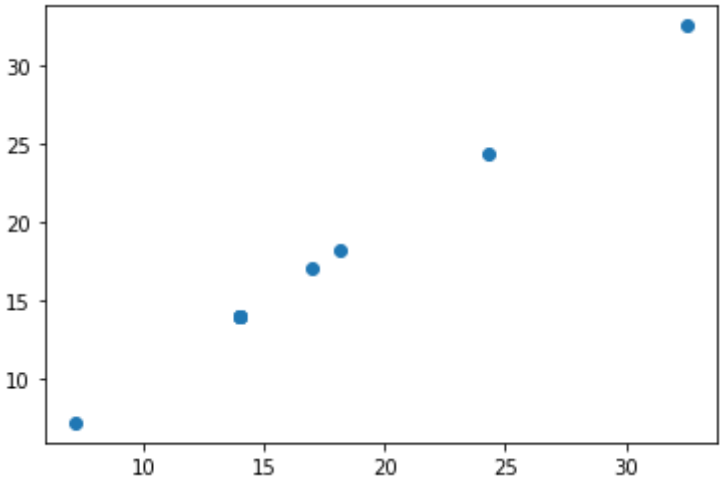
Out[23]:

	Co-effecient
Birth Rate	1.0

Best Fit line

In [24]:
```
prediction=lr.predict(g_test)
plt.scatter(h_test,prediction)
```

Out[24]:
```
<matplotlib.collections.PathCollection at 0x2e3302089d0>
```



To find score

In [25]:
```
print(lr.score(g_test,h_test))
```

```
1.0
```

Import Lasso and ridge

In [26]:
```
from sklearn.linear_model import Ridge,Lasso
```

Ridge

In [27]:
```
ri=Ridge(alpha=5)
ri.fit(g_train,h_train)
```

Out[27]:
```
Ridge(alpha=5)
```

In [28]:
```
ri.score(g_test,h_test)
```

Out[28]:
```
0.9999503294747978
```

In [29]:
```
ri.score(g_train,h_train)
```

Out[29]: 0.9999505291894991

Lasso

In [30]:
```
l=Lasso(alpha=6)
l.fit(g_train,h_train)
```

Out[30]: Lasso(alpha=6)

In [31]:
```
l.score(g_test,h_test)
```

Out[31]: 0.9973884725557933

In [32]:
```
ri.score(g_train,h_train)
```

Out[32]: 0.9999505291894991