Deena 20104016

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

Problem Statement

LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("18_world-data-2023.csv")
a
```

Out[2]:

	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	
...	...	...	...	...	...	...	...	...	
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

HEAD

In [3]:
```
a.head()
```

Out[3]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural	Land	Armed	Birth	Calling	Cap

				Land(%)	Area(Km2)	Forces size	Rate	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

5 rows × 35 columns

Data Cleaning and Preprocessing

In [30]:
```
d=a.head(10)
d
```

Out[30]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Cap
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	Bu
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 rows × 35 columns

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

Out[5]:

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Latit

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Latit
count	189.000000	194.000000	188.000000	189.000000	187.000000	181.000000	188.000000	194.000
mean	20.214974	360.546392	2.698138	21.332804	72.279679	160.392265	1.839840	19.092
std	9.945774	323.236419	1.282267	19.548058	7.483661	233.502024	1.684261	23.961
min	5.900000	1.000000	0.980000	1.400000	52.800000	2.000000	0.010000	-40.900
25%	11.300000	82.500000	1.705000	6.000000	67.000000	13.000000	0.332500	4.544
50%	17.950000	255.500000	2.245000	14.000000	73.200000	53.000000	1.460000	17.273

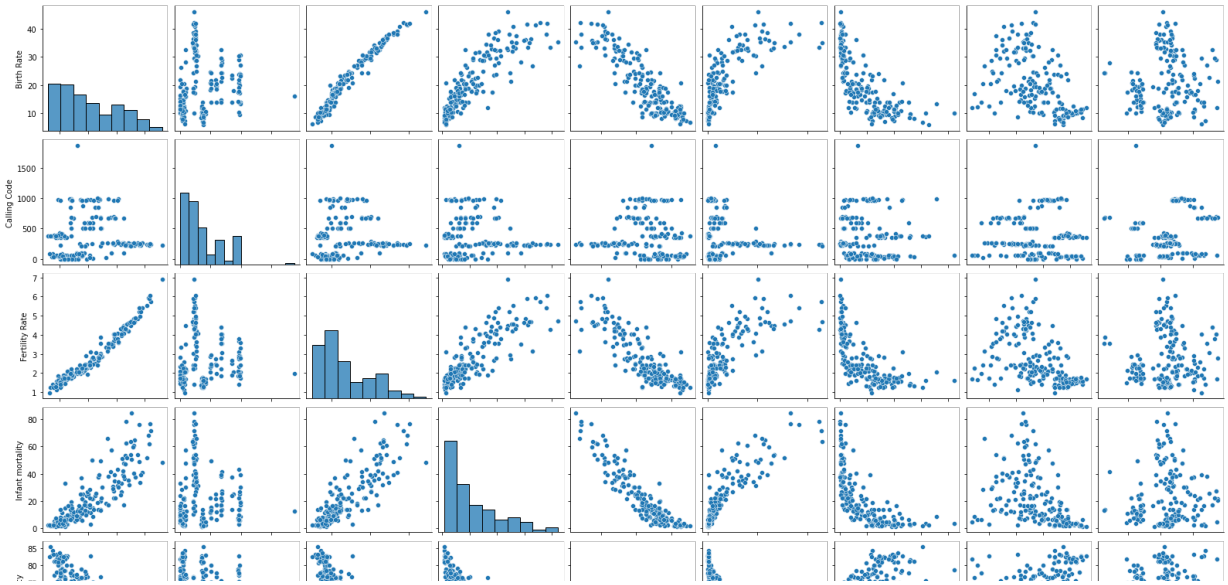
To display heading

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')
```

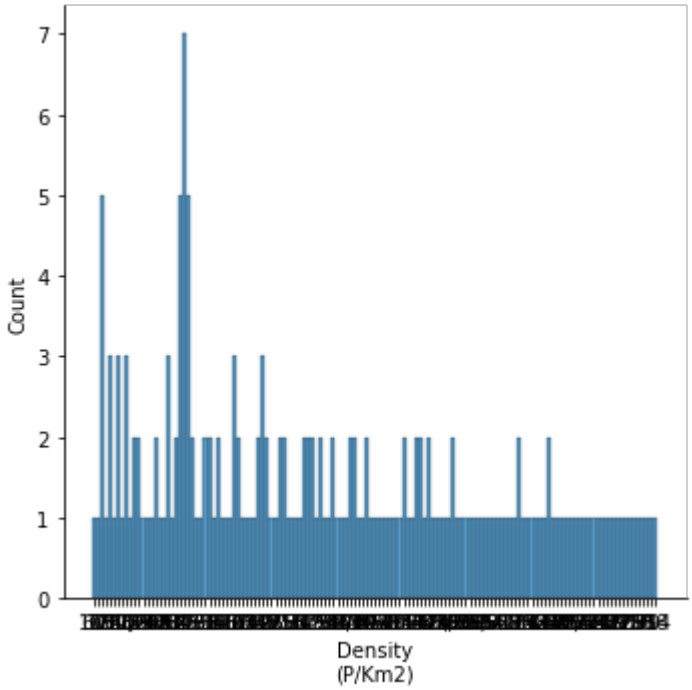
In [7]:
```
sns.pairplot(a)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x143e2befdf0>
```



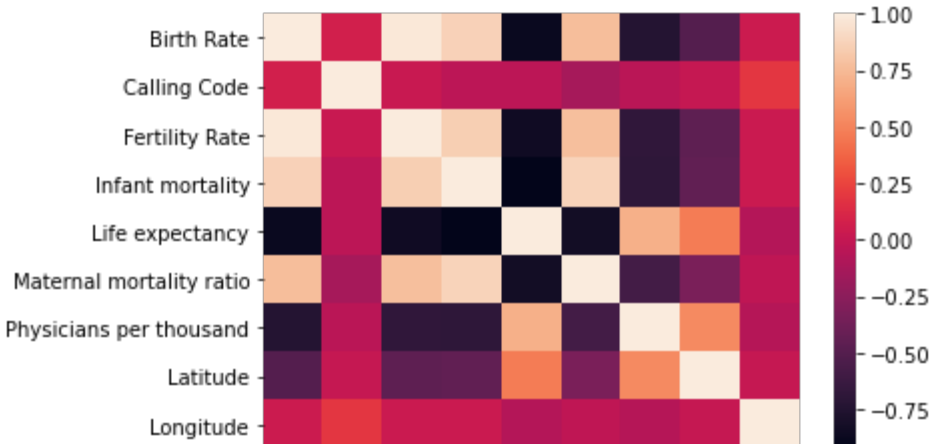
In [8]:
```
sns.displot(a['Density\n(P/Km2)'])
```

Out[8]: <seaborn.axisgrid.FacetGrid at 0x143e403eb20>



In [9]:
```
sns.heatmap(a.corr())
```

Out[9]: <AxesSubplot:>



TO TRAIN THE MODEL - MODEL BUILDING

In [31]:
```
x = d[['Birth Rate']]
y = d['Birth Rate']
```

In [32]:
```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [33]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[33]:
```
LinearRegression()
```

In [34]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[34]:

	Co-efficient
Birth Rate	1.0

In [35]:
```
prediction= lr.predict(x_test)
pp.scatter(y_test,prediction)
```

Out[35]:
```
<matplotlib.collections.PathCollection at 0x143e803fe50>
```

In [36]:
```
lr.score(x_test,y_test)
```

Out[36]: 1.0

RIDGE & LASSO

In [37]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[37]: Ridge(alpha=10)

In [38]:
```
rr.score(x_test,y_test)
```

Out[38]: 0.9998544326854026

In [39]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[39]: Lasso(alpha=10)

In [40]:
```
la.score(x_test,y_test)
```

Out[40]: 0.9927103369905057