In [1]: # import libaries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

In [24]: x=pd.read_csv(r"C:\Users\user\Downloads\18_world-data-2023 - 18_world-data-202

Out[24]:

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

195 rows × 35 columns

In [25]: x=x.head(10)

Out[25]:

| | 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 | |
| 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 rows × 35 columns

```
In [26]:
```

<class 'pandas.core.frame.DataFrame'> RangeIndex: 10 entries, 0 to 9 Data columns (total 35 columns): # Column Non-Null Count Dtype ----------0 Country 10 non-null object 1 Density (P/Km2)10 non-null object Abbreviation 10 non-null object 2 3 Agricultural Land(%) 10 non-null object Land Area(Km2) 4 object 10 non-null 5 Armed Forces size 9 non-null object 6 Birth Rate 10 non-null float64 7 Calling Code 10 non-null float64 8 Capital/Major City 10 non-null object 9 Co2-Emissions 10 non-null object 10 CPI 9 non-null object 11 CPI Change (%) 9 non-null object 12 Currency-Code 10 non-null object Fertility Rate 13 10 non-null float64 14 Forested Area (%) 10 non-null object 15 Gasoline Price 10 non-null object **GDP** object 16 10 non-null 17 Gross primary education enrollment (%) 10 non-null object Gross tertiary education enrollment (%) 18 9 non-null object float64 Infant mortality 19 10 non-null Largest city object 20 10 non-null 21 Life expectancy 9 non-null float64 Maternal mortality ratio 9 non-null float64 23 Minimum wage 9 non-null object Official language object 24 10 non-null Out of pocket health expenditure object 10 non-null 26 Physicians per thousand 10 non-null float64 27 Population 10 non-null object 28 Population: Labor force participation (%) object 8 non-null 29 Tax revenue (%) 9 non-null object 30 Total tax rate 9 non-null object Unemployment rate object 31 8 non-null 32 Urban_population 10 non-null object 33 Latitude 10 non-null float64 34 Longitude 10 non-null float64 dtypes: float64(9), object(26) memory usage: 2.9+ KB

In [50]: d=x[['Co2-Emissions','Birth Rate','Fertility Rate']]

Out[50]:

| _ | | Co2-Emissions | Birth Rate | Fertility Rate | | |
|---|---|---------------|------------|----------------|--|--|
| | 0 | 8,672 | 32.49 | 4.47 | | |
| | 1 | 4,536 | 11.78 | 1.62 | | |
| | 2 | 150,006 | 24.28 | 3.02 | | |
| | 3 | 469 | 7.20 | 1.27 | | |
| | 4 | 34,693 | 40.73 | 5.52 | | |
| | 5 | 557 | 15.33 | 1.99 | | |
| | 6 | 201,348 | 17.02 | 2.26 | | |
| | 7 | 5,156 | 13.99 | 1.76 | | |
| | 8 | 375,908 | 12.60 | 1.74 | | |
| | 9 | 61,448 | 9.70 | 1.47 | | |

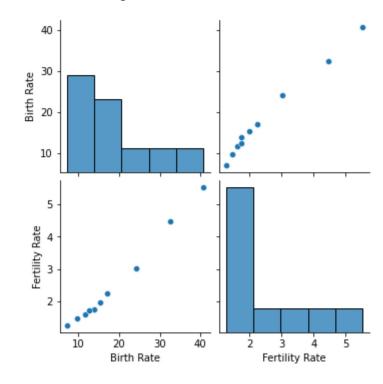
In [51]:

Out[51]:

| Birth Rate | Calling Code | Fertility Rate | Infant mortality | Life expectancy | Maternal mortality ratio | Physicians per thousand | Latit |
|---------------|--|---|--|---|--|---|---|
| 10.000000 | 10.000000 | 10.000000 | 10.000000 | 9.000000 | 9.000000 | 10.000000 | 10.000 |
| 18.512000 | 181.400000 | 2.512000 | 16.090000 | 74.788889 | 124.888889 | 2.671000 | 17.538 |
| 10.754729 | 149.167467 | 1.416622 | 18.504321 | 7.376897 | 206.621904 | 1.738387 | 31.192 |
| 7.200000 | 1.000000 | 1.270000 | 2.700000 | 60.800000 | 5.000000 | 0.210000 | -38.416 |
| 11.985000 | 55.750000 | 1.650000 | 3.575000 | 74.900000 | 15.000000 | 1.330000 | -4.136 |
| 14.660000 | 153.000000 | 1.875000 | 8.300000 | 76.700000 | 39.000000 | 3.045000 | 30.986 |
| 22.465000 | 327.250000 | 2.830000 | 17.825000 | 78.500000 | 112.000000 | 3.890000 | 40.882 |
| 40.730000 | 376.000000 | 5.520000 | 51.600000 | 82.700000 | 638.000000 | 5.170000 | 47.516 |
| | Rate 10.000000 18.512000 10.754729 7.200000 11.985000 14.660000 22.465000 | Rate Code 10.000000 10.000000 18.512000 181.400000 10.754729 149.167467 7.200000 1.000000 11.985000 55.750000 14.660000 153.000000 22.465000 327.250000 | Rate Code Rate 10.000000 10.000000 10.000000 18.512000 181.400000 2.512000 10.754729 149.167467 1.416622 7.200000 1.000000 1.270000 11.985000 55.750000 1.650000 14.660000 153.000000 1.875000 22.465000 327.250000 2.830000 | Rate Code Rate mortality 10.000000 10.000000 10.000000 10.000000 18.512000 181.400000 2.512000 16.090000 10.754729 149.167467 1.416622 18.504321 7.200000 1.000000 1.270000 2.700000 11.985000 55.750000 1.650000 3.575000 14.660000 153.000000 1.875000 8.300000 22.465000 327.250000 2.830000 17.825000 | Rate Code Rate mortality expectancy 10.000000 10.000000 10.000000 9.000000 18.512000 181.400000 2.512000 16.090000 74.788889 10.754729 149.167467 1.416622 18.504321 7.376897 7.200000 1.000000 1.270000 2.700000 60.800000 11.985000 55.750000 1.650000 3.575000 74.900000 14.660000 153.000000 1.875000 8.300000 76.700000 22.465000 327.250000 2.830000 17.825000 78.500000 | Birth Rate Code Code Fertility Rate Infant mortality Life expectancy mortality ratio 10.000000 10.000000 10.000000 9.000000 9.000000 9.000000 18.512000 181.400000 2.512000 16.090000 74.788889 124.888889 10.754729 149.167467 1.416622 18.504321 7.376897 206.621904 7.200000 1.000000 1.270000 2.700000 60.800000 5.000000 11.985000 55.750000 1.650000 3.575000 74.900000 15.000000 14.660000 153.000000 1.875000 8.300000 76.700000 39.000000 22.465000 327.250000 2.830000 17.825000 78.500000 112.000000 | Rate Code Rate Infant mortality Rate Elfe mortality mortality ratio mortality ratio per thousand 10.000000 10.000000 10.000000 10.000000 9.000000 9.000000 10.000000 18.512000 181.400000 2.512000 16.090000 74.788889 124.888889 2.671000 10.754729 149.167467 1.416622 18.504321 7.376897 206.621904 1.738387 7.200000 1.000000 1.270000 2.700000 60.800000 5.000000 0.210000 11.985000 55.750000 1.650000 3.575000 74.900000 15.000000 1.330000 14.660000 153.000000 2.830000 76.700000 39.000000 3.890000 22.465000 327.250000 2.830000 17.825000 78.500000 112.000000 3.890000 |

In [52]:

Out[52]: <seaborn.axisgrid.PairGrid at 0x1866aa27640>

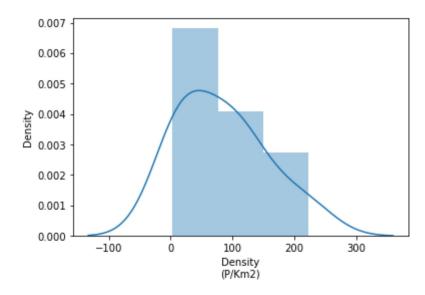


In [53]:

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut ureWarning: `distplot` is a deprecated function and will be removed in a futu re version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for hi stograms).

warnings.warn(msg, FutureWarning)

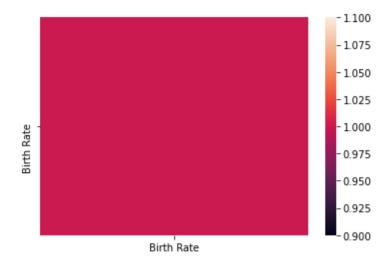
Out[53]: <AxesSubplot:xlabel='Density\n(P/Km2)', ylabel='Density'>



In [54]:

In [55]:

Out[55]: <AxesSubplot:>



In [63]: x=x1[['Birth Rate']]

```
In [64]: # to split my dataset into traning and test date
         from sklearn.model_selection import train_test_split
In [65]: from sklearn.linear_model import LinearRegression
         lr=LinearRegression()
Out[65]: LinearRegression()
In [66]:
          -3.552713678800501e-15
         coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
Out[67]:
                    Co-efficient
           Birth Rate
                           1.0
In [68]: prediction=lr.predict(x_test)
Out[68]: <matplotlib.collections.PathCollection at 0x1866afce9d0>
           15.0
           14.5
           14.0
           13.5
           13.0
           12.5
              12.5
                     13.0
                             13.5
                                     14.0
                                             14.5
                                                    15.0
In [69]: L
Out[69]: 1.0
In [70]: __
Out[70]: 1.0
In [71]: -
```

```
In [72]: rr=Ridge(alpha=10)
    rr.fit(x_train,y_train)
Out[72]: 0.9962114080235698

In [73]: la=Lasso(alpha=10)
Out[73]: Lasso(alpha=10)
In [74]:
Out[74]: 0.8104258940992788

In []:
```