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

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

Out[2]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capi
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

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

Out[3]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital
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	An
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	

5 rows × 35 columns

DATA CLEANING AND DATA PREPROCESSING

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 195 entries, 0 to 194
Data columns (total 35 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Country                                       195 non-null    object
 1   Density
(P/Km2)                               195 non-null    object
 2   Abbreviation                                  188 non-null    object
 3   Agricultural Land( %)                         188 non-null    object
 4   Land Area(Km2)                                194 non-null    object
 5   Armed Forces size                             171 non-null    object
 6   Birth Rate                                    189 non-null    float64
 7   Calling Code                                  194 non-null    float64
 8   Capital/Major City                            192 non-null    object
 9   Co2-Emissions                                 188 non-null    object
 10  CPI                                           178 non-null    object
 11  CPI Change (%)                                179 non-null    object
 12  Currency-Code                                 180 non-null    object
 13  Fertility Rate                                188 non-null    float64
 14  Forested Area (%)                             188 non-null    object
 15  Gasoline Price                                175 non-null    object
 16  GDP                                           193 non-null    object
 17  Gross primary education enrollment (%)        188 non-null    object
 18  Gross tertiary education enrollment (%)       183 non-null    object
 19  Infant mortality                              189 non-null    float64
 20  Largest city                                  189 non-null    object
 21  Life expectancy                               187 non-null    float64
 22  Maternal mortality ratio                      181 non-null    float64
 23  Minimum wage                                  150 non-null    object
 24  Official language                             194 non-null    object
 25  Out of pocket health expenditure              188 non-null    object
 26  Physicians per thousand                       188 non-null    float64
 27  Population                                    194 non-null    object
 28  Population: Labor force participation (%)     176 non-null    object
 29  Tax revenue (%)                               169 non-null    object
 30  Total tax rate                                183 non-null    object
 31  Unemployment rate                             176 non-null    object
 32  Urban_population                              190 non-null    object
 33  Latitude                                      194 non-null    float64
 34  Longitude                                     194 non-null    float64
dtypes: float64(9), object(26)
memory usage: 53.4+ KB
```

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

Out[5]:

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Latitude
count	189.000000	194.000000	188.000000	189.000000	187.000000	181.000000	188.000000	194.000000
mean	20.214974	360.546392	2.698138	21.332804	72.279679	160.392265	1.839840	19.092351

	Birth Rate	Calling Code	Fertility Rate	Infant mortality	Life expectancy	Maternal mortality ratio	Physicians per thousand	Latitude
std	9.945774	323.236419	1.282267	19.548058	7.483661	233.502024	1.684261	23.96177
min	5.900000	1.000000	0.980000	1.400000	52.800000	2.000000	0.010000	-40.90055
25%	11.300000	82.500000	1.705000	6.000000	67.000000	13.000000	0.332500	4.54417
50%	17.950000	255.500000	2.245000	14.000000	73.200000	53.000000	1.460000	17.27384
75%	28.750000	506.750000	3.597500	32.700000	77.500000	186.000000	2.935000	40.12460
max	46.080000	1876.000000	6.910000	84.500000	85.400000	1150.000000	8.420000	64.96305

In [6]:
```
df.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]:
```
df1=df.dropna()
df1
```

Out[7]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Ca
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	
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	
6	Argentina	17	AR	54.30%	2,780,400	105,000	17.02	54.0	E
...	...	...	...	...	...	...	...	...	
185	United Kingdom	281	GB	71.70%	243,610	148,000	11.00	44.0	
186	United States	36	US	44.40%	9,833,517	1,359,000	11.60	1.0	
187	Uruguay	20	UY	82.60%	176,215	22,000	13.86	598.0	
191	Vietnam	314	VN	39.30%	331,210	522,000	16.75	84.0	

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Ca
193	Zambia	25	ZM	32.10%	752,618	16,000	36.19	260.0	

110 rows × 35 columns

In [8]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 110 entries, 0 to 193
Data columns (total 35 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Country                                       110 non-null    object
 1   Density
(P/Km2)                                 110 non-null    object
 2   Abbreviation                                  110 non-null    object
 3   Agricultural Land( %)                         110 non-null    object
 4   Land Area(Km2)                                110 non-null    object
 5   Armed Forces size                             110 non-null    object
 6   Birth Rate                                    110 non-null    float64
 7   Calling Code                                  110 non-null    float64
 8   Capital/Major City                            110 non-null    object
 9   Co2-Emissions                                 110 non-null    object
 10  CPI                                           110 non-null    object
 11  CPI Change (%)                                110 non-null    object
 12  Currency-Code                                 110 non-null    object
 13  Fertility Rate                                110 non-null    float64
 14  Forested Area (%)                             110 non-null    object
 15  Gasoline Price                                110 non-null    object
 16  GDP                                           110 non-null    object
 17  Gross primary education enrollment (%)        110 non-null    object
 18  Gross tertiary education enrollment (%)       110 non-null    object
 19  Infant mortality                              110 non-null    float64
 20  Largest city                                  110 non-null    object
 21  Life expectancy                               110 non-null    float64
 22  Maternal mortality ratio                      110 non-null    float64
 23  Minimum wage                                  110 non-null    object
 24  Official language                             110 non-null    object
 25  Out of pocket health expenditure              110 non-null    object
 26  Physicians per thousand                       110 non-null    float64
 27  Population                                    110 non-null    object
 28  Population: Labor force participation (%)     110 non-null    object
 29  Tax revenue (%)                               110 non-null    object
 30  Total tax rate                                110 non-null    object
 31  Unemployment rate                             110 non-null    object
 32  Urban_population                              110 non-null    object
 33  Latitude                                      110 non-null    float64
 34  Longitude                                     110 non-null    float64
dtypes: float64(9), object(26)
memory usage: 30.9+ KB
```

In [9]:
```
df1.columns
```

Out[9]:
```
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 (%)',
```

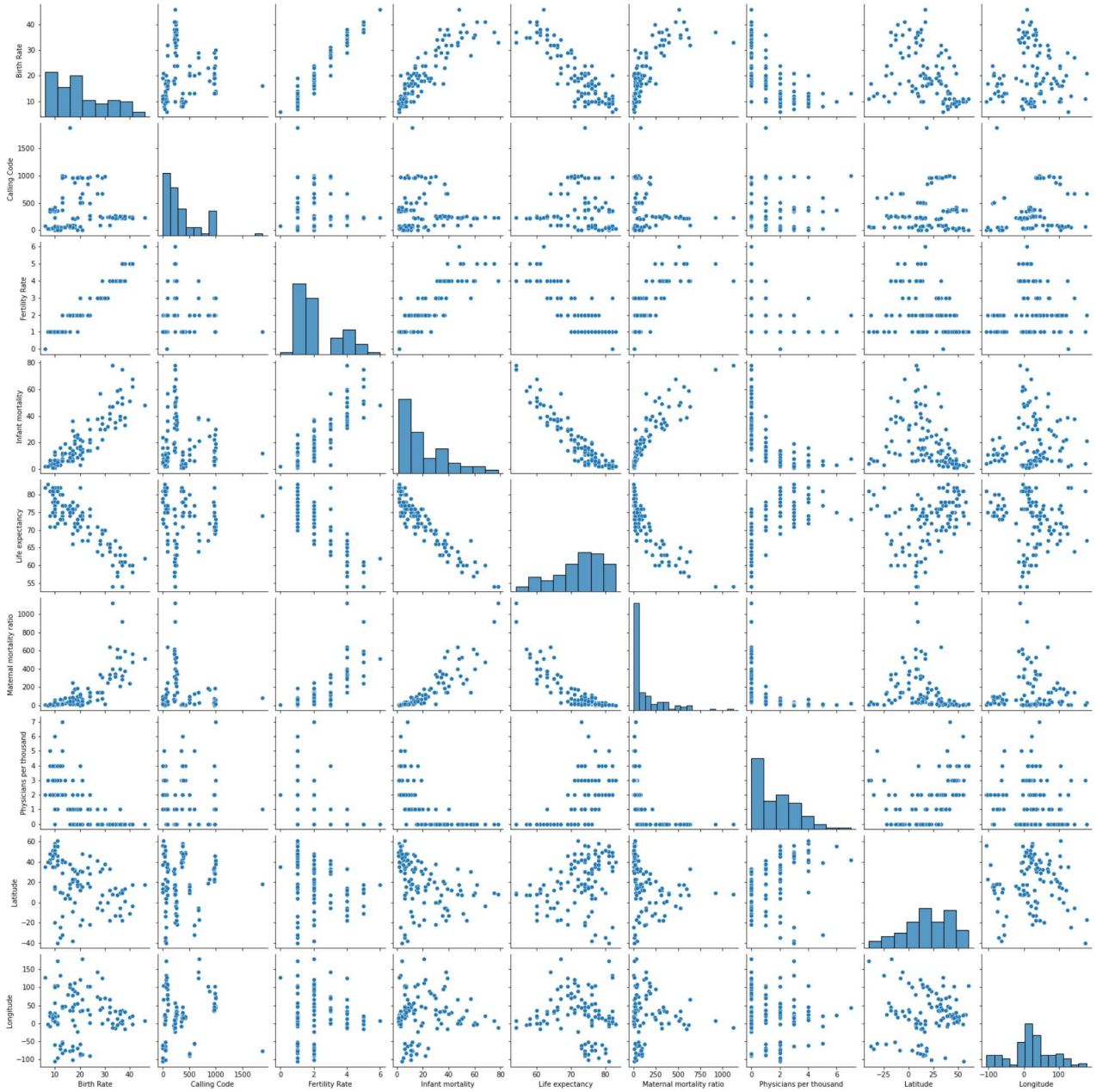
```
       'Currency-Code', 'Fertility Rate', 'Forested Area (%)',
       'Gasoline Price', 'GDP', 'Gross primary education enrollment (%)',
       'Gross tertiary education enrollment (%)', 'Infant mortality',
       'Largest city', 'Life expectancy', 'Maternal mortality ratio',
       'Minimum wage', '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 [10]:
```
df1=df1[[ 'Birth Rate', 'Calling Code','Fertility Rate','Infant mortality','Life expect
```

EDA AND VISUALIZATION

In [11]:
```
sns.pairplot(df1)
```

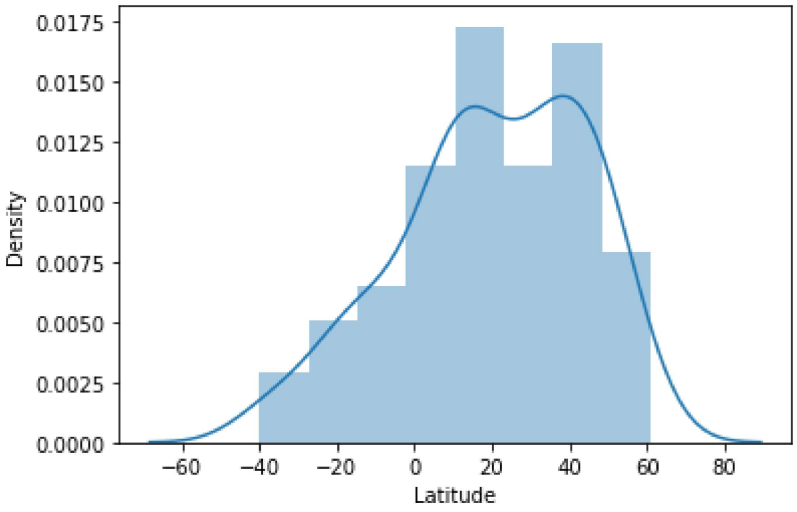
Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x1f1f4b359d0>
```



In [12]:
```
sns.distplot(df1[ 'Latitude'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[12]: <AxesSubplot:xlabel='Latitude', ylabel='Density'>



In [13]:
```
sns.heatmap(df1.corr())
```

Out[13]: <AxesSubplot:>

TO TRAIN THE MODEL AND MODEL BULDING

In [14]:
```
df2=df[['Calling Code','Fertility Rate', 'Infant mortality', 'Life expectancy', 'Matern
       'Physicians per thousand', 'Latitude',
       'Longitude','Birth Rate']].dropna()
df2=df2[df['Calling Code']!="NaN"]
df2=df2[df['Fertility Rate']!="NaN"]
df2=df2[df['Infant mortality']!="NaN"]
df2=df2[df['Life expectancy']!="NaN"]
df2=df2[df['Maternal mortality ratio']!="NaN"]
df2=df2[df['Physicians per thousand']!="NaN"]
df2=df2[df['Latitude']!="NaN"]
```

```
df2=df2[df['Longitude']!="NaN"]
df2=df2[df['Birth Rate']!="NaN"]
x=df2[['Calling Code','Fertility Rate', 'Infant mortality', 'Life expectancy', 'Materna
       'Physicians per thousand', 'Latitude',
       'Longitude']]
y=df2['Birth Rate']
```

```
<ipython-input-14-de6015c771a4>:4: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Calling Code']!="NaN"]
<ipython-input-14-de6015c771a4>:5: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Fertility Rate']!="NaN"]
<ipython-input-14-de6015c771a4>:6: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Infant mortality']!="NaN"]
<ipython-input-14-de6015c771a4>:7: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Life expectancy']!="NaN"]
<ipython-input-14-de6015c771a4>:8: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Maternal mortality ratio']!="NaN"]
<ipython-input-14-de6015c771a4>:9: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Physicians per thousand']!="NaN"]
<ipython-input-14-de6015c771a4>:10: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Latitude']!="NaN"]
<ipython-input-14-de6015c771a4>:11: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Longitude']!="NaN"]
<ipython-input-14-de6015c771a4>:12: UserWarning: Boolean Series key will be reindexed to
match DataFrame index.
  df2=df2[df['Birth Rate']!="NaN"]
```

In [15]:
```
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 [16]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[16]:
```
LinearRegression()
```

In [17]:
```
lr.intercept_
```

Out[17]:
```
15.062696928967153
```

In [18]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

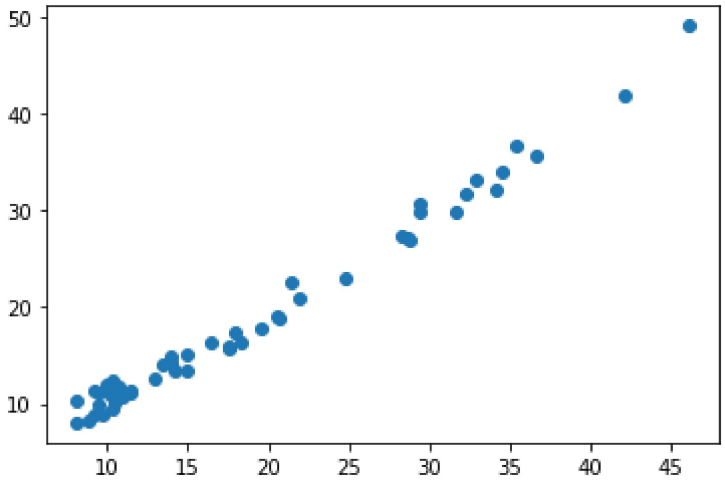
Out[18]:

	Co-efficient
Calling Code	0.001082
Fertility Rate	6.324806

	Co-efficient
Infant mortality	0.024906
Life expectancy	-0.155382
Maternal mortality ratio	-0.002730
Physicians per thousand	-0.510924
Latitude	-0.005534
Longitude	0.000084

In [19]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[19]: <matplotlib.collections.PathCollection at 0x1f1f9a73970>



ACCURACY

In [20]:
```
lr.score(x_test,y_test)
```

Out[20]: 0.9841138543645098

In [21]:
```
lr.score(x_train,y_train)
```

Out[21]: 0.9747727692934772

In [22]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [23]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[23]: Ridge(alpha=10)

In [24]:
```
rr.score(x_test,y_test)
```

Out[24]: 0.9800019713946799

In [25]:
```
rr.score(x_train,y_train)
```

Out[25]: 0.9699370373722661

In [26]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[26]: Lasso(alpha=10)

In [27]:
```
la.score(x_test,y_test)
```

Out[27]: 0.7932721305875804

In [28]:
```
la.score(x_train,y_train)
```

Out[28]: 0.7750841082869347