31-07-2023

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

In [345]:
```
a=pd.read_csv(r"C:\Users\user\Downloads\18_world-data-2023.csv")
a
```

Out[345]:

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

195 rows × 35 columns

In [346]:
```
a=a.head(10)
a
```

Out[346]:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital/Major City
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0	Kabu
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0	Tirana
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0	Algiers
3	Andorra	164	AD	40.00%	468	NaN	7.20	376.0	Andorra la Vella
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	Luanda
5	Antigua and Barbuda	223	AG	20.50%	443	0	15.33	1.0	St. John's, Saint John
6	Argentina	17	AR	54.30%	2,780,400	105,000	17.02	54.0	Buenos Aires
7	Armenia	104	AM	58.90%	29,743	49,000	13.99	374.0	Yerevan
8	Australia	3	AU	48.20%	7,741,220	58,000	12.60	61.0	Canberra
9	Austria	109	AT	32.40%	83,871	21,000	9.70	43.0	Vienna

10 rows × 35 columns

In [347]:
```
a.info()
```

```
<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
 2   Abbreviation                                10 non-null     object
 3   Agricultural Land( %)                       10 non-null     object
 4   Land Area(Km2)                              10 non-null     object
 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
 13  Fertility Rate                              10 non-null     float64
 14  Forested Area (%)                           10 non-null     object
 15  Gasoline Price                              10 non-null     object
 16  GDP                                         10 non-null     object
 17  Gross primary education enrollment (%)      10 non-null     object
 18  Gross tertiary education enrollment (%)     9 non-null      object
 19  Infant mortality                            10 non-null     float64
 20  Largest city                                10 non-null     object
 21  Life expectancy                             9 non-null      float64
 22  Maternal mortality ratio                    9 non-null      float64
 23  Minimum wage                                9 non-null      object
 24  Official language                           10 non-null     object
 25  Out of pocket health expenditure            10 non-null     object
 26  Physicians per thousand                     10 non-null     float64
 27  Population                                  10 non-null     object
 28  Population: Labor force participation (%)   8 non-null      object
 29  Tax revenue (%)                             9 non-null      object
 30  Total tax rate                              9 non-null      object
 31  Unemployment rate                           8 non-null      object
 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 [348]:
```
a.columns
```

Out[348]:
```
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 [349]:
```
d=a[['Country', 'Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land( %)',
       'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Code']]
d
```

Out[349]:

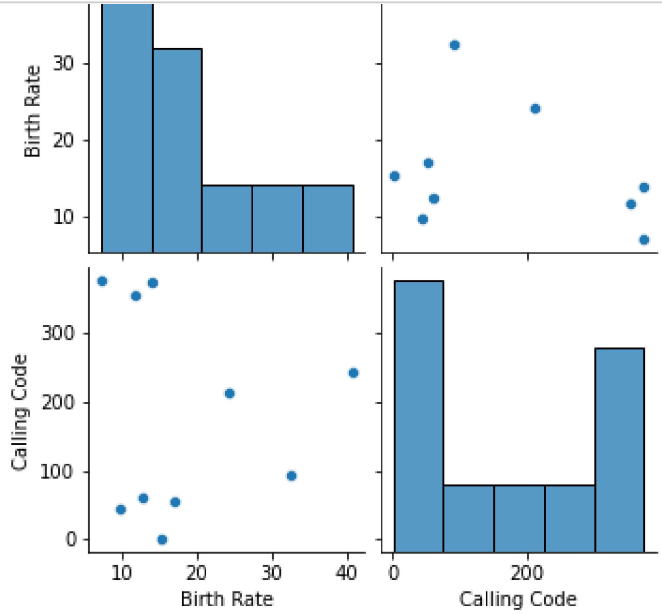
	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0
3	Andorra	164	AD	40.00%	468	NaN	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

In [360]:
```
d.describe()
```

Out[360]:

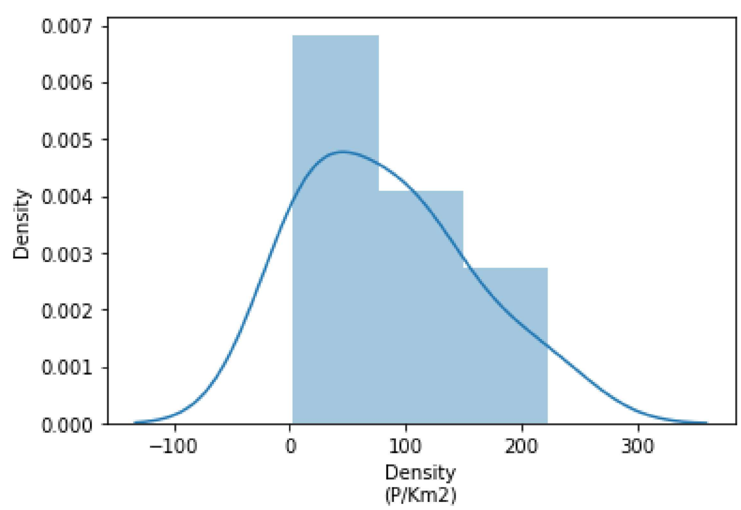
	Birth Rate	Calling Code
count	10.000000	10.000000
mean	18.512000	181.400000
std	10.754729	149.167467
min	7.200000	1.000000
25%	11.985000	55.750000
50%	14.660000	153.000000
75%	22.465000	327.250000
max	40.730000	376.000000

In [351]:
```
sns.pairplot(d)
```



In [353]:
```
sns.distplot(a['Density\n(P/Km2)'])
```

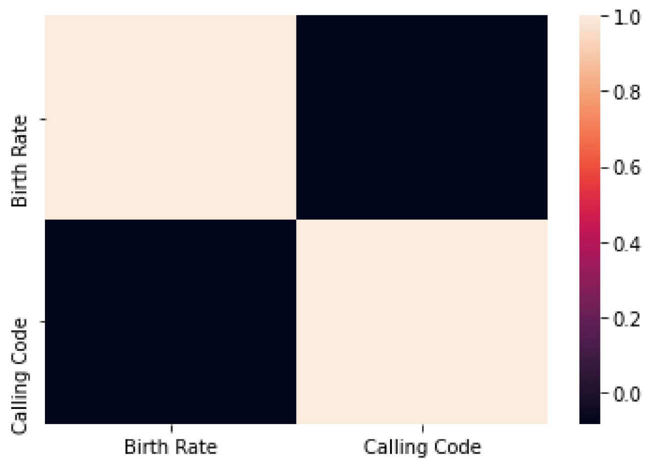
Out[353]: <AxesSubplot:xlabel='Density\n(P/Km2)', ylabel='Density'>



In [361]:
```
x1=a[[ 'Birth Rate', 'Calling Code']]
```

In [362]:
```
sns.heatmap(x1.corr())
```

Out[362]: <AxesSubplot:>



In [363]:
```
x=a[['Birth Rate', 'Calling Code']]
y=a['Density\n(P/Km2)']
```

In [364]:
```
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 [365]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[365]: LinearRegression()

In [366]:
```
print(lr.intercept_)
```

154.69810898570245

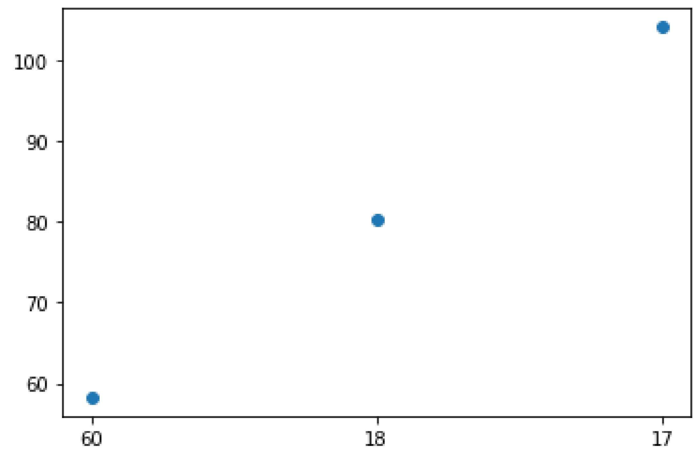
In [367]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[367]:

	Co-efficient
Birth Rate	-2.919609
Calling Code	-0.016401

In [368]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[368]:
```
<matplotlib.collections.PathCollection at 0x190c47edf10>
```



In [369]:
```
print(lr.score(x_test,y_test))
```

```
-8.526473654816035
```

In [370]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [371]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[371]:
```
Ridge(alpha=10)
```

In [372]:
```
rr.score(x_test,y_test)
```

Out[372]:
```
-8.564611639176906
```

In [373]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[373]:
```
Lasso(alpha=10)
```

In [374]:
```
la.score(x_test,y_test)
```

Out[374]:
```
-8.611623528648572
```

In [375]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[375]:
```
ElasticNet()
```

In [376]:
```
print(en.coef_)
```

```
[-2.90170038 -0.01639415]
```

In [377]:
```
print(en.intercept_)
```

```
154.41191458285985
```

In [378]:
```
print(en.predict(x_test))
```

```
[ 58.61101308  80.46667475 104.13968989]
```

In [379]:
```
en.score(x_test,y_test)
```

Out[379]:
```
-8.544001349703986
```

In [380]:
```
from sklearn import metrics
```

In [381]:
```
print("Mean Absolute Error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error 50.37420723327395
```

In [382]:
```
print("Mean Squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 3825.408420945017
```

In [383]:
```
print(" Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(y_test,prediction))
```

```
 Root Mean Squared Error 61.84988618376769
```

In [384]:
```
import pickle
```

In [385]:
```
filename="prediction"
pickle.dump(lr,open(filename,'wb'))
```

In [386]:
```
import pandas as pd
import pickle
```

In [387]:
```
filename="prediction"
model=pickle.load(open(filename,"rb"))
```

In [388]:
```
real=[[10,20],[15,30]]
result=model.predict(real)
```

In [389]:
```
result
```

Out[389]:
```
array([125.17400621, 110.41195483])
```