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

In [2]:
```
data=pd.read_csv(r"C:\Users\user\Downloads\20_states - 20_states.csv")
data
```

Out[2]:

	id	name	country_id	country_code	country_name	state_code	type	latitude
0	3901	Badakhshan	1	AF	Afghanistan	BDS	NaN	36.734772
1	3871	Badghis	1	AF	Afghanistan	BDG	NaN	35.167134
2	3875	Baghlan	1	AF	Afghanistan	BGL	NaN	36.178903
3	3884	Balkh	1	AF	Afghanistan	BAL	NaN	36.755060
4	3872	Bamyan	1	AF	Afghanistan	BAM	NaN	34.810007
...	...	...	...	...	...	...	...	...
5072	1953	Mashonaland West Province	247	ZW	Zimbabwe	MW	NaN	-17.485103
5073	1960	Masvingo Province	247	ZW	Zimbabwe	MV	NaN	-20.624151
5074	1954	Matabeleland North Province	247	ZW	Zimbabwe	MN	NaN	-18.533157
5075	1952	Matabeleland South Province	247	ZW	Zimbabwe	MS	NaN	-21.052337
5076	1957	Midlands Province	247	ZW	Zimbabwe	MI	NaN	-19.055201

5077 rows × 9 columns

In [3]:
```
df=data.head(100)
df
```

Out[3]:

	id	name	country_id	country_code	country_name	state_code	type	latitude	lo
0	3901	Badakhshan	1	AF	Afghanistan	BDS	NaN	36.734772	70
1	3871	Badghis	1	AF	Afghanistan	BDG	NaN	35.167134	63
2	3875	Baghlan	1	AF	Afghanistan	BGL	NaN	36.178903	68
3	3884	Balkh	1	AF	Afghanistan	BAL	NaN	36.755060	66
4	3872	Bamyan	1	AF	Afghanistan	BAM	NaN	34.810007	67
...	...	...	...	...	...	...	...	...	
95	1105	Chlef	4	DZ	Algeria	2	NaN	36.169351	1
96	1121	Constantine	4	DZ	Algeria	25	NaN	36.337391	6
97	4912	Djanet	4	DZ	Algeria	56	NaN	23.831087	8
98	1098	Djelfa	4	DZ	Algeria	17	NaN	34.670396	3
99	1129	El Bayadh	4	DZ	Algeria	32	NaN	32.714882	0

100 rows × 9 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   id            100 non-null    int64  
 1   name          100 non-null    object 
 2   country_id    100 non-null    int64  
 3   country_code  100 non-null    object 
 4   country_name  100 non-null    object 
 5   state_code    100 non-null    object 
 6   type          0 non-null      object 
 7   latitude      100 non-null    float64
 8   longitude     100 non-null    float64
dtypes: float64(2), int64(2), object(5)
memory usage: 7.2+ KB
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['id', 'name', 'country_id', 'country_code', 'country_name',
       'state_code', 'type', 'latitude', 'longitude'],
      dtype='object')
```

In [6]:
```
x=df[['id', 'country_id','latitude']]
y=df['longitude']
```

In [7]:
```
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 [8]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[8]:
```
LinearRegression()
```

In [9]:
```
print(lr.intercept_)
```

```
188.20640431703944
```

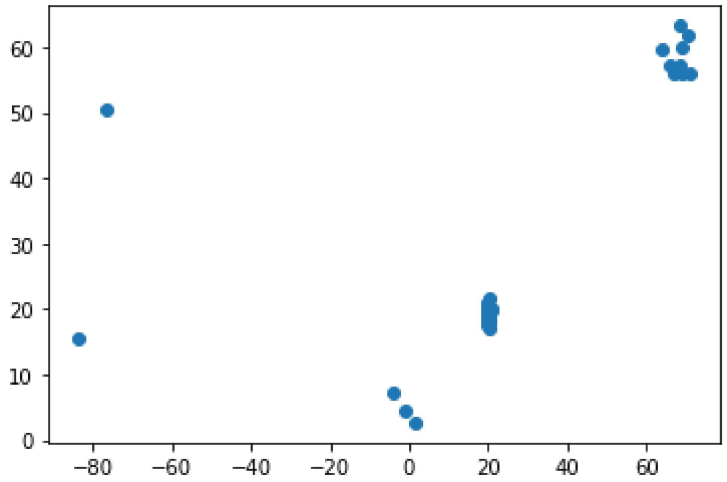
In [10]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[10]:

	Co-efficient
id	-0.006581
country_id	-24.323027
latitude	-2.239640

In [11]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[11]:
```
<matplotlib.collections.PathCollection at 0x1812f5a4df0>
```



In [12]:
```
print(lr.score(x_test,y_test))
```

```
0.34557989731176086
```

In [13]:
```
print(lr.score(x_train,y_train))
```

```
0.6707827927637978
```

In [14]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [15]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[15]:
```
Ridge(alpha=10)
```

In [16]:
```
rr.score(x_test,y_test)
```

Out[16]:
```
0.2927719760168427
```

In [17]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[17]:
```
Lasso(alpha=10)
```

In [18]:
```
la.score(x_test,y_test)
```

Out[18]:
```
0.16537454347109348
```

In [19]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[19]:
```
ElasticNet()
```

In [20]:
```
print(en.coef_)
```

```
[ 7.79621195e-03 -8.75510466e+00  6.21087362e-01]
```

In [21]:
```
print(en.intercept_)
```

```
14.0168051185506
```

In [22]:
```
print(en.predict(x_test))
```

```
[18.44926877 56.26745273 18.29848759 57.34308414 18.33746865 17.95687914
 31.52429515 19.00562952 17.38083279 56.75028782 58.47856599 58.37029113
 17.73551907 58.04367384 18.08526112 18.74435097 17.85301743 17.58170596
 58.49022624 57.28269935  9.66766666 18.35450978 59.86450191 10.07552781
 17.94372326 17.88869537 57.94228096 18.31937655 18.34303911 17.87941984]
```

In [23]:
```
print(en.score(x_test,y_test))
```

```
0.23302776988904683
```

Evaluation metrics

In [24]:
```
from sklearn import metrics
```

In [25]:
```
print("Mean absolute error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean absolute error 11.706157871783757
```

In [26]:
```
print("Mean squared error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean squared error 901.7252513529937
```

In [27]:
```
print("Mean squared error",np.sqrt(metrics.mean_squared_error(y_test,predictic
```

```
Mean squared error 30.028740422351945
```

Model Saving

In [28]:
```
import pickle
```

In [29]:
```
filename='prediction'
pickle.dump(lr,open(filename,'wb'))
```

In [ ]:
```

```