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\21_cities - 21_cities.csv")
data
```

Out[2]:

	id	name	state_id	state_code	state_name	country_id	country_code	countr
0	52	Ashkāsham	3901	BDS	Badakhshan	1	AF	Afgł
1	68	Fayzabad	3901	BDS	Badakhshan	1	AF	Afgł
2	78	Jurm	3901	BDS	Badakhshan	1	AF	Afgł
3	84	Khandūd	3901	BDS	Badakhshan	1	AF	Afgł
4	115	Rāghistān	3901	BDS	Badakhshan	1	AF	Afgł
...	...	...	...	...	...	...	...	
150449	131496	Redcliff	1957	MI	Midlands Province	247	ZW	Zir
150450	131502	Shangani	1957	MI	Midlands Province	247	ZW	Zir
150451	131503	Shurugwi	1957	MI	Midlands Province	247	ZW	Zir
150452	131504	Shurugwi District	1957	MI	Midlands Province	247	ZW	Zir
150453	131508	Zvishavane District	1957	MI	Midlands Province	247	ZW	Zir

150454 rows × 11 columns

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

Out[3]:

	id	name	state_id	state_code	state_name	country_id	country_code	country_name
0	52	Ashkāsham	3901	BDS	Badakhshan	1	AF	Afghanistan
1	68	Fayzabad	3901	BDS	Badakhshan	1	AF	Afghanistan
2	78	Jurm	3901	BDS	Badakhshan	1	AF	Afghanistan
3	84	Khandūd	3901	BDS	Badakhshan	1	AF	Afghanistan
4	115	Rāghistān	3901	BDS	Badakhshan	1	AF	Afghanistan
...	...	...	...	...	...	...	...	...
95	180	Bashkia Poliçan	629	BR	Berat District	3	AL	Albania
96	186	Bashkia Skrapar	629	BR	Berat District	3	AL	Albania
97	191	Berat	629	BR	Berat District	3	AL	Albania
98	280	Çorovodë	629	BR	Berat District	3	AL	Albania
99	219	Kuçovë	629	BR	Berat District	3	AL	Albania

100 rows × 11 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   id            100 non-null    int64  
 1   name          100 non-null    object 
 2   state_id      100 non-null    int64  
 3   state_code    100 non-null    object 
 4   state_name    100 non-null    object 
 5   country_id    100 non-null    int64  
 6   country_code  100 non-null    object 
 7   country_name  100 non-null    object 
 8   latitude      100 non-null    float64
 9   longitude     100 non-null    float64
 10  wikiDataId    100 non-null    object 
dtypes: float64(2), int64(3), object(6)
memory usage: 8.7+ KB
```

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

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

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

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_)
```

```
-3.765876499528531e-13
```

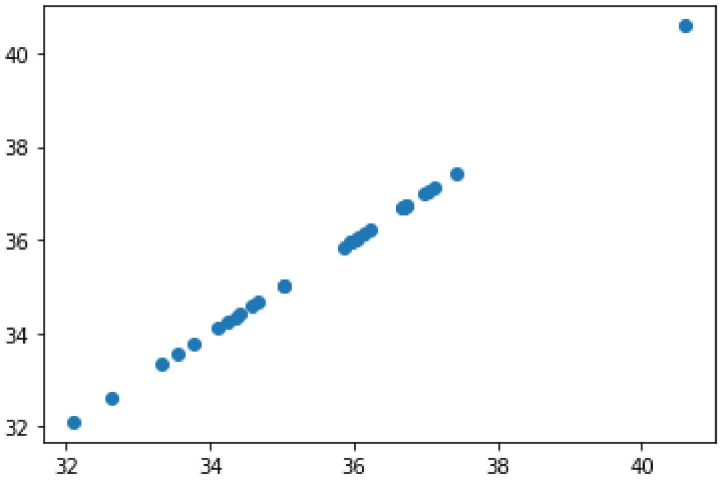
In [10]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[10]:

	Co-efficient
id	4.602989e-18
state_id	7.914676e-17
country_id	1.161222e-13
latitude	1.000000e+00
longitude	-5.139643e-16

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

Out[11]: <matplotlib.collections.PathCollection at 0x209983aafd0>



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

```
1.0
```

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

```
1.0
```

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]: 0.9976329265007883

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]: 0.056756764675745974

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

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

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

```
[-0.00164638 -0.00094384  0.          0.72049145  0.02575576]
```

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

```
11.770182131069216
```

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

```
[35.69848184 34.31021039 35.63260119 36.14810055 40.63836254 36.57709354
 35.71153497 32.78946491 35.82535888 34.61386997 36.27505233 36.5363532
 36.17901807 34.13159904 35.00204544 34.725074   33.95681083 35.50378974
 33.93814851 36.32708931 36.16208892 33.7884018  33.08834566 36.14727426
 34.597915   35.60398412 35.45126896 34.96373557 34.51107679 36.36798057]
```

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

```
0.9338044979260387
```

Evaluation metrics

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

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

```
Mean absolute error 4.736951571734001e-16
```

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

```
Mean squared error 3.365806528942984e-30
```

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

```
Mean squared error 1.8346134549116833e-15
```

Model Saving

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

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

In [ ]:
```

```