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

In [4]:
```
df=pd.read_csv(r"C:\Users\Admin\Downloads\21_cities.csv")
df
```

Out[4]:

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

150454 rows × 11 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150454 entries, 0 to 150453
Data columns (total 11 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   id            150454 non-null  int64  
 1   name          150454 non-null  object 
 2   state_id      150454 non-null  int64  
 3   state_code    150129 non-null  object 
 4   state_name    150454 non-null  object 
 5   country_id    150454 non-null  int64  
 6   country_code  150406 non-null  object 
 7   country_name  150454 non-null  object 
 8   latitude      150454 non-null  float64
 9   longitude     150454 non-null  float64
 10  wikiDataId    147198 non-null  object 
dtypes: float64(2), int64(3), object(6)
memory usage: 12.6+ MB
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['id', 'name', 'state_id', 'state_code', 'state_name', 'country_id',
       'country_code', 'country_name', 'latitude', 'longitude', 'wikiDataI
d'],
      dtype='object')
```

In [7]:
```
df1=df.head(100)
```

In [8]:
```
df1.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 [9]:
```
x=df1[['state_id', 'country_id','latitude']]
y=df1[['longitude']]
```

In [10]:
```
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 [11]:
```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

Out[11]:
```
LinearRegression()
```

In [12]:
```
print('Linear Regresion(score):',lr.score(x_test,y_test))
print('Linear Regression(train score)',lr.score(x_train,y_train))
```

```
Linear Regresion(score): 0.9807952040111171
Linear Regression(train score) 0.9257581704223419
```

In [13]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[13]:
```
Ridge(alpha=10)
```

In [14]:
```
print('Ridge(test score):',rr.score(x_test,y_test))
```

```
Ridge(test score): 0.9806792416936592
```

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

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

In [16]:
```
print('Lasso (test score)',la.score(x_test,y_test))
```

```
Lasso (test score) 0.97371122886158
```

In [17]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[17]:
```
ElasticNet()
```

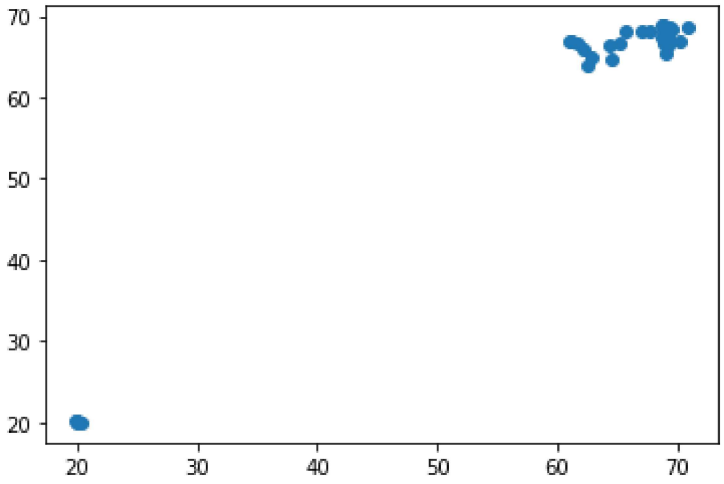
In [18]:
```
print(en.score(x_test,y_test))
```

```
0.9793617406524431
```

In [19]:
```
import pickle
file="predict"
pickle.dump(lr,open(file,'wb'))
```

In [20]:
```
prediction= lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[20]: <matplotlib.collections.PathCollection at 0x2d453b6fa60>



In [ ]: