D18

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

In [10]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\21_cities.csv")
df
```

Out[10]:

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

150454 rows × 11 columns

In [11]:
```
df.head(10)
```

Out[11]:

	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
5	131	Wākhān	3901	BDS	Badakhshan	1	AF	Afghanistan
6	72	Ghormach	3871	BDG	Badghis	1	AF	Afghanistan
7	108	Qala i Naw	3871	BDG	Badghis	1	AF	Afghanistan
8	54	Baghlān	3875	BGL	Baghlan	1	AF	Afghanistan
9	140	Ḩukūmatī Dahanah-ye Ghōrī	3875	BGL	Baghlan	1	AF	Afghanistan

In [12]:
```
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 [13]:
```
dff=df.dropna()
```

In [14]:
```
dff.describe()
```

Out[14]:

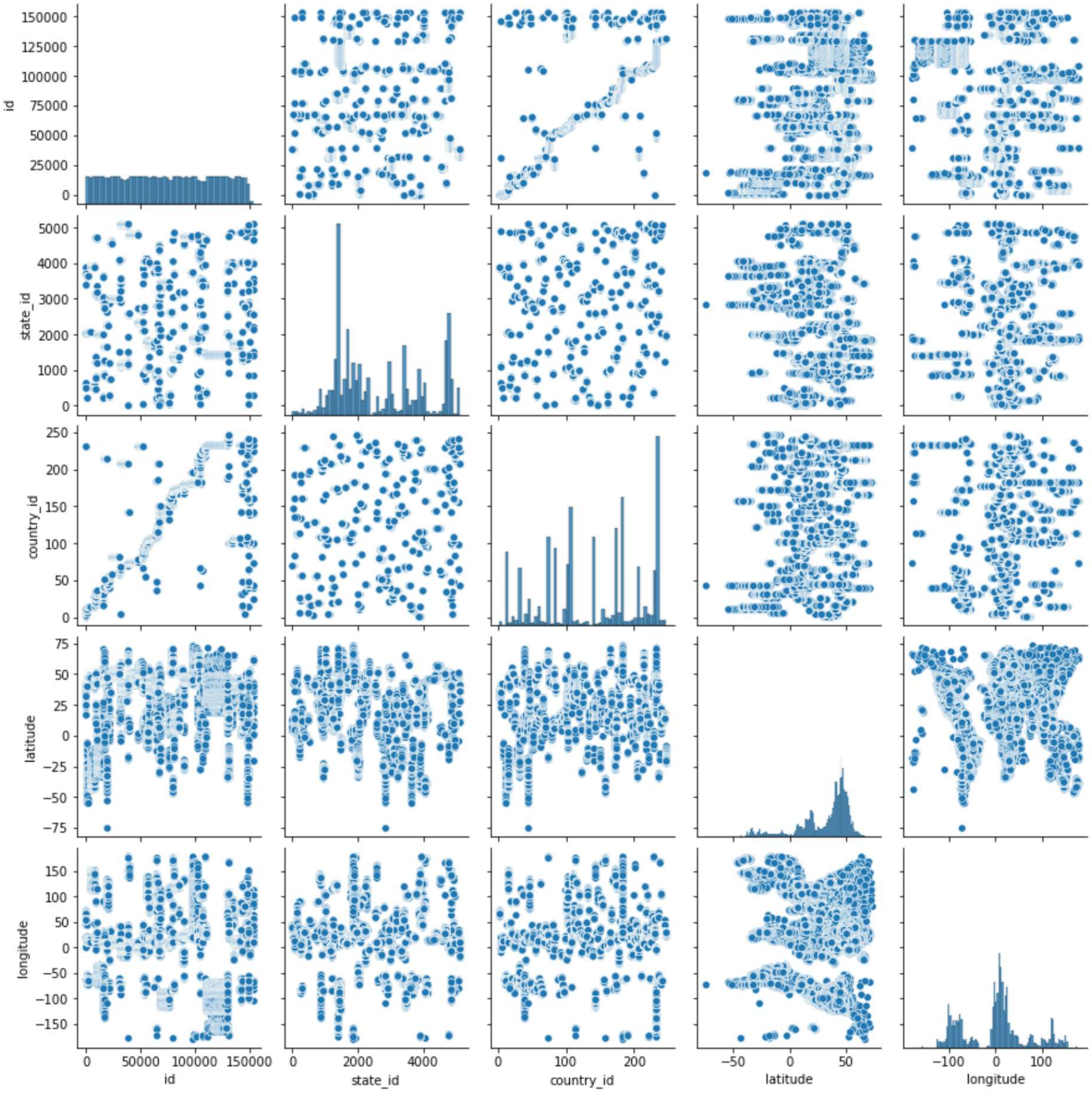
	id	state_id	country_id	latitude	longitude
count	146959.000000	146959.000000	146959.000000	146959.000000	146959.000000
mean	74815.628618	2664.711124	139.865929	32.037738	2.043877
std	43408.845239	1351.363226	70.916362	22.701273	68.681166
min	1.000000	1.000000	1.000000	-75.000000	-179.121980
25%	37448.500000	1452.000000	82.000000	19.521795	-61.751760
50%	74428.000000	2131.000000	142.000000	40.896880	8.837560
75%	112849.500000	3904.000000	207.000000	47.349195	27.623705
max	153528.000000	5116.000000	247.000000	73.508190	179.466000

In [15]:
```
dff.columns
```

Out[15]:
```
Index(['id', 'name', 'state_id', 'state_code', 'state_name', 'country_id',
       'country_code', 'country_name', 'latitude', 'longitude', 'wikiDataI
d'],
      dtype='object')
```

```
In [16]: sns.pairplot(dff)
```

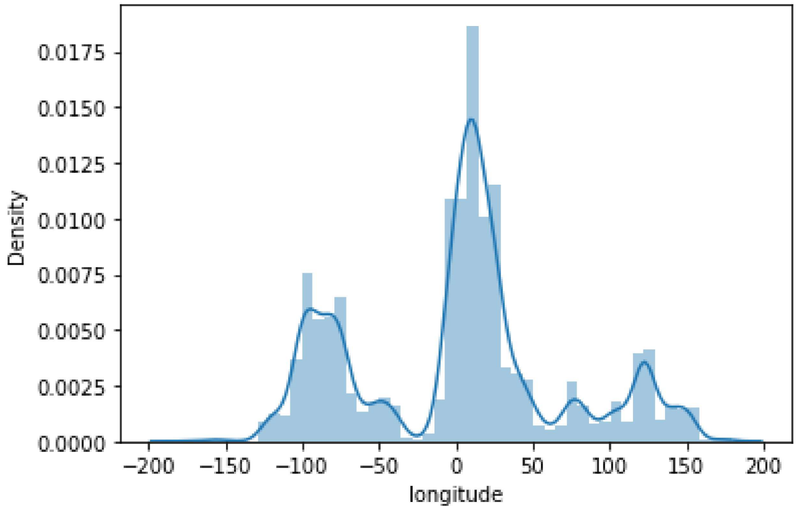
```
Out[16]: <seaborn.axisgrid.PairGrid at 0x200742a23d0>
```



In [18]:
```
sns.distplot(dff['longitude'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

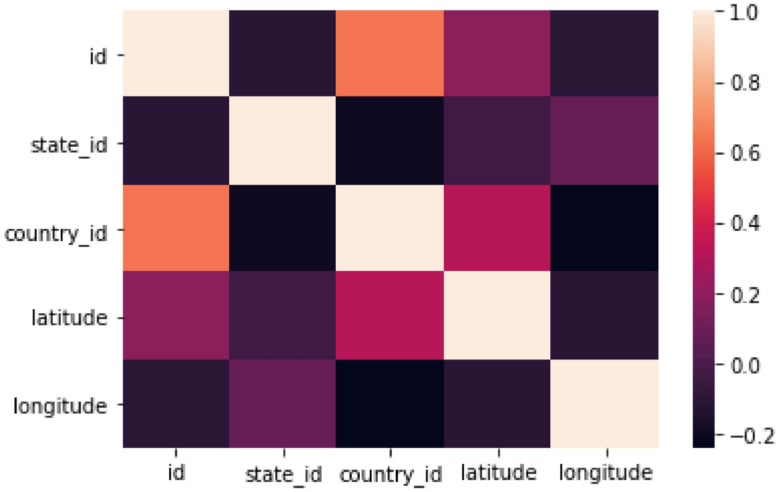
Out[18]:
```
<AxesSubplot:xlabel='longitude', ylabel='Density'>
```



In [19]:
```
df1=dff[['id', 'state_id', 'country_id', 'latitude', 'longitude']]
```

In [20]:
```
sns.heatmap(df1.corr())
```

Out[20]:
```
<AxesSubplot:>
```



In [22]:
```
x=df1[['id', 'state_id', 'country_id', 'latitude']]
y=df1['longitude']
```

In [23]:
```
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 [24]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[24]:
```
LinearRegression()
```

In [25]:
```
print(lr.intercept_)
```

```
26.554827312825697
```

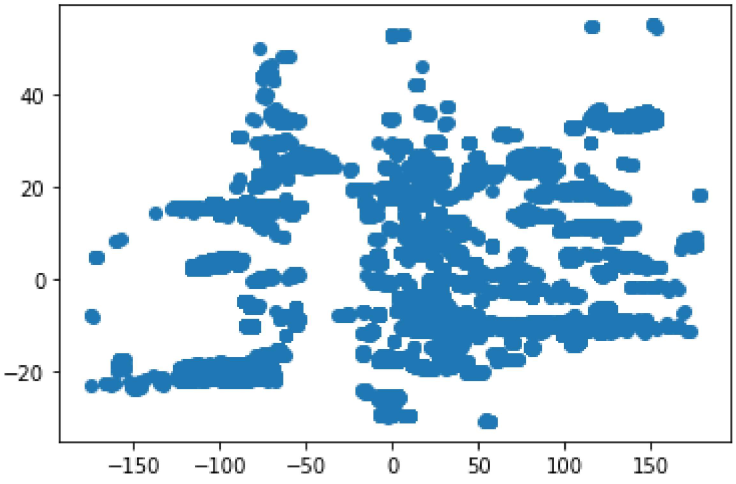
In [26]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[26]:

	Co-efficient
id	0.000137
state_id	0.002126
country_id	-0.268305
latitude	-0.095711

In [27]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[27]:
```
<matplotlib.collections.PathCollection at 0x2007a04cfa0>
```



In [28]:
```
print(lr.score(x_test,y_test))
```

```
0.06430737704117473
```

In [29]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [30]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[30]:
```
Ridge(alpha=10)
```

In [31]:
```
rr.score(x_test,y_test)
```

Out[31]:
```
0.06430737699183875
```

In [32]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[32]:
```
Lasso(alpha=10)
```

In [33]:
```
la.score(x_test,y_test)
```

Out[33]:
```
0.06424871179274771
```

In [34]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[34]:
```
ElasticNet()
```

In [35]:
```
print(en.coef_)
```

```
[ 1.37161394e-04  2.12680729e-03 -2.68203659e-01 -9.46727499e-02]
```

In [36]:
```
print(en.intercept_)
```

```
26.520863096742467
```

In [37]:
```
print(en.predict(x_train))
```

```
[ -3.61407417  12.96329353 -21.18210423 ...  -6.54037993 -29.26722618
   3.96603652]
```

In [38]:
```
print(en.score(x_train,y_train))
```

```
0.0647900381273504
```

In [39]:
```
from sklearn import metrics
```

In [40]:
```
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolytre Error: 52.200460709817506
```

In [41]:
```
print("Mean Square Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Square Error: 4392.054654493994
```

In [42]:

```
print("Root Mean Square Error:",np.sqrt(metrics.mean_absolute_error(y_test,pre
```

```
Root Mean Square Error: 7.224988630428252
```

In [43]:

```
import pickle
```

In [44]:

```
f3="prediction"
pickle.dump(lr,open(f3,'wb'))
```

In [ ]: