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

In [32]:
```
s=pd.read_csv(r"C:\Users\user\Downloads\19_nuclear_explosions - 19_nuclear_explosions.csv")
s
```

Out[32]:

	WEAPON SOURCE COUNTRY	WEAPON DEPLOYMENT LOCATION	Data.Source	Location.Cordinates.Latitude	Location.Cordinates.Longitude	Data.Magnitude.Body	Data.Magnitude.Surface	Locatior
0	USA	Alamogordo	DOE	32.54	-105.57	0.0	0.0	
1	USA	Hiroshima	DOE	34.23	132.27	0.0	0.0	
2	USA	Nagasaki	DOE	32.45	129.52	0.0	0.0	
3	USA	Bikini	DOE	11.35	165.20	0.0	0.0	
4	USA	Bikini	DOE	11.35	165.20	0.0	0.0	
...	...	...	...	...	...	...	...	
2041	CHINA	Lop Nor	HFS	41.69	88.35	5.3	0.0	
2042	INDIA	Pokhran	HFS	27.07	71.70	5.3	0.0	
2043	INDIA	Pokhran	NRD	27.07	71.70	0.0	0.0	
2044	PAKIST	Chagai	HFS	28.90	64.89	0.0	0.0	
2045	PAKIST	Kharan	HFS	28.49	63.78	5.0	0.0	

2046 rows × 16 columns

In [33]:
```
s=s.head(20)
s
```

Out[33]:

	WEAPON SOURCE COUNTRY	WEAPON DEPLOYMENT LOCATION	Data.Source	Location.Cordinates.Latitude	Location.Cordinates.Longitude	Data.Magnitude.Body	Data.Magnitude.Surface	Location.C
0	USA	Alamogordo	DOE	32.54	-105.57	0.0	0.0	
1	USA	Hiroshima	DOE	34.23	132.27	0.0	0.0	
2	USA	Nagasaki	DOE	32.45	129.52	0.0	0.0	
3	USA	Bikini	DOE	11.35	165.20	0.0	0.0	
4	USA	Bikini	DOE	11.35	165.20	0.0	0.0	
5	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
6	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
7	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
8	USSR	Semi Kazakh	DOE	48.00	76.00	0.0	0.0	
9	USA	Nts	DOE	37.00	-116.00	0.0	0.0	
10	USA	Nts	DOE	37.00	-116.00	0.0	0.0	
11	USA	Nts	DOE	37.00	-116.00	0.0	0.0	
12	USA	Nts	DOE	37.00	-116.00	0.0	0.0	
13	USA	Nts	DOE	37.00	-116.00	0.0	0.0	
14	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
15	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
16	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
17	USA	Enewetak	DOE	11.30	162.15	0.0	0.0	
18	USSR	Semi Kazakh	DOE	48.00	76.00	0.0	0.0	
19	USSR	Semi Kazakh	DOE	48.00	76.00	0.0	0.0	

In [34]:
```
s.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   WEAPON SOURCE COUNTRY           20 non-null     object 
 1   WEAPON DEPLOYMENT LOCATION      20 non-null     object 
 2   Data.Source                     20 non-null     object 
 3   Location.Cordinates.Latitude    20 non-null     float64
 4   Location.Cordinates.Longitude   20 non-null     float64
 5   Data.Magnitude.Body             20 non-null     float64
 6   Data.Magnitude.Surface          20 non-null     float64
 7   Location.Cordinates.Depth       20 non-null     float64
 8   Data.Yeild.Lower                20 non-null     float64
 9   Data.Yeild.Upper                20 non-null     float64
 10  Data.Purpose                    20 non-null     object 
 11  Data.Name                       20 non-null     object 
 12  Data.Type                       20 non-null     object 
 13  Date.Day                        20 non-null     int64  
 14  Date.Month                      20 non-null     int64  
 15  Date.Year                       20 non-null     int64  
dtypes: float64(7), int64(3), object(6)
memory usage: 2.6+ KB
```

In [35]:
```
s.describe()
```

Out[35]:

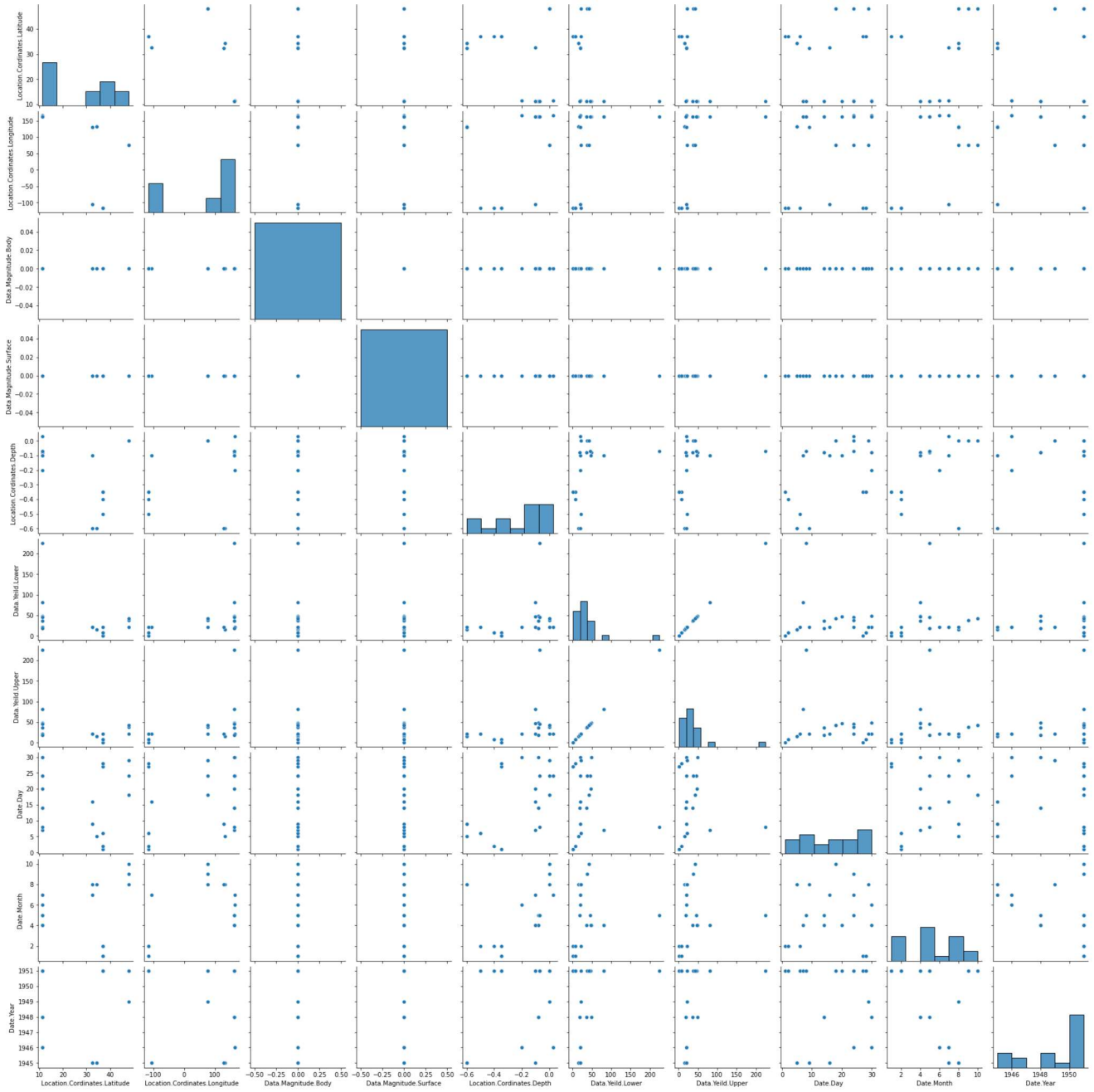
	Location.Cordinates.Latitude	Location.Cordinates.Longitude	Data.Magnitude.Body	Data.Magnitude.Surface	Location.Cordinates.Depth	Data.Yeild.Lower	Da
count	20.000000	20.000000	20.0	20.0	20.000000	20.000000	
mean	26.501000	63.483500	0.0	0.0	-0.200000	37.175000	
std	14.771883	123.128335	0.0	0.0	0.204862	48.239882	
min	11.300000	-116.000000	0.0	0.0	-0.600000	1.000000	
25%	11.300000	-108.177500	0.0	0.0	-0.350000	17.250000	
50%	32.495000	130.895000	0.0	0.0	-0.100000	21.500000	
75%	37.000000	162.150000	0.0	0.0	-0.070000	42.875000	
max	48.000000	165.200000	0.0	0.0	0.030000	225.000000	

In [36]:
```
s.columns
```

Out[36]:
```
Index(['WEAPON SOURCE COUNTRY', 'WEAPON DEPLOYMENT LOCATION', 'Data.Source',
       'Location.Cordinates.Latitude', 'Location.Cordinates.Longitude',
       'Data.Magnitude.Body', 'Data.Magnitude.Surface',
       'Location.Cordinates.Depth', 'Data.Yeild.Lower', 'Data.Yeild.Upper',
       'Data.Purpose', 'Data.Name', 'Data.Type', 'Date.Day', 'Date.Month',
       'Date.Year'],
      dtype='object')
```

In [38]:
```
sns.pairplot(s)
```

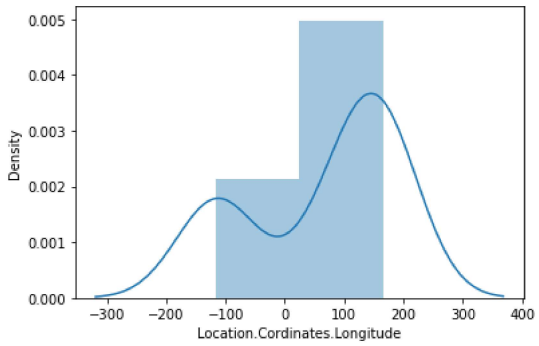
Out[38]: <seaborn.axisgrid.PairGrid at 0x23cee83ebe0>



In [40]:
```
sns.distplot(s['Location.Cordinates.Longitude'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function an
d will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar fle
xibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[40]: <AxesSubplot:xlabel='Location.Cordinates.Longitude', ylabel='Density'>



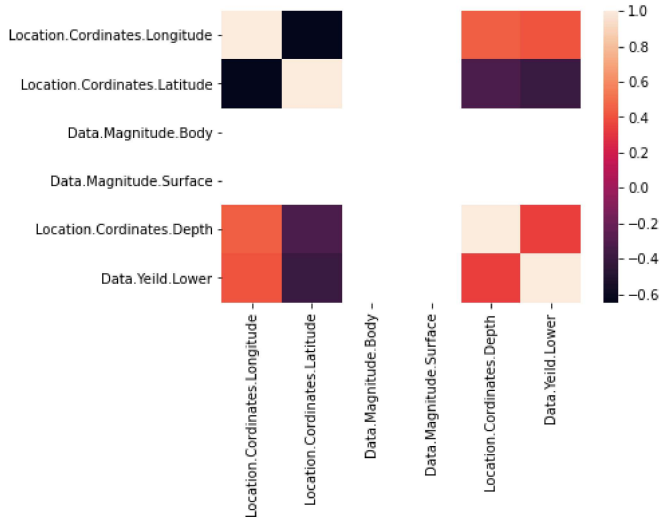
In [61]:
```
ude','Location.Cordinates.Latitude','Data.Magnitude.Body','Data.Magnitude.Surface','Location.Cordinates.Depth','Data.Yeild.Lower'
```

Out[61]:

	Location.Cordinates.Longitude	Location.Cordinates.Latitude	Data.Magnitude.Body	Data.Magnitude.Surface	Location.Cordinates.Depth	Data.Yeild.Lower
0	-105.57	32.54	0.0	0.0	-0.10	21.0
1	132.27	34.23	0.0	0.0	-0.60	15.0
2	129.52	32.45	0.0	0.0	-0.60	21.0
3	165.20	11.35	0.0	0.0	-0.20	21.0
4	165.20	11.35	0.0	0.0	0.03	21.0
5	162.15	11.30	0.0	0.0	-0.08	37.0
6	162.15	11.30	0.0	0.0	-0.08	49.0
7	162.15	11.30	0.0	0.0	-0.08	18.0
8	76.00	48.00	0.0	0.0	0.00	22.0
9	-116.00	37.00	0.0	0.0	-0.35	1.0
10	-116.00	37.00	0.0	0.0	-0.35	8.0
11	-116.00	37.00	0.0	0.0	-0.35	1.0
12	-116.00	37.00	0.0	0.0	-0.40	8.0
13	-116.00	37.00	0.0	0.0	-0.50	22.0
14	162.15	11.30	0.0	0.0	-0.10	81.0
15	162.15	11.30	0.0	0.0	-0.10	47.0
16	162.15	11.30	0.0	0.0	-0.07	225.0
17	162.15	11.30	0.0	0.0	-0.07	45.5
18	76.00	48.00	0.0	0.0	0.00	38.0
19	76.00	48.00	0.0	0.0	0.00	42.0

In [62]:
```
sns.heatmap(s1.corr())
```

Out[62]: <AxesSubplot:>



In [63]:
```
x=s1[['Location.Cordinates.Longitude','Location.Cordinates.Latitude','Data.Magnitude.Body','Data.Magnitude.Surface']]
y=s1['Data.Magnitude.Body']
```

In [64]:
```
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 [65]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[65]: LinearRegression()

In [66]:
```
lr.intercept_
```

Out[66]: 0.0

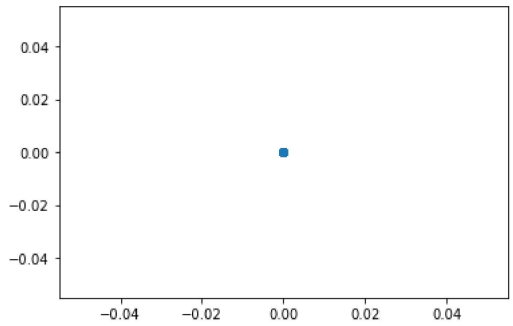
In [67]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[67]:

	Co-efficient
Location.Cordinates.Longitude	0.0
Location.Cordinates.Latitude	0.0
Data.Magnitude.Body	0.0
Data.Magnitude.Surface	0.0

In [68]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[68]: <matplotlib.collections.PathCollection at 0x23cf6b861f0>



In [69]:
```
print(lr.score(x_test,y_test))
```

```
1.0
```

In [70]:
```
from sklearn.linear_model import Ridge,Lasso
from sklearn.linear_model import Ridge,Lasso
```

In [71]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

Out[71]: 1.0

In [72]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
la.score(x_test,y_test)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:530: ConvergenceWarning: Objective did n
ot converge. You might want to increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[72]: 1.0

In [73]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:530: ConvergenceWarning: Objective did n
ot converge. You might want to increase the number of iterations. Duality gap: 0.0, tolerance: 0.0
  model = cd_fast.enet_coordinate_descent(
```

Out[73]: ElasticNet()

In [74]:
```
print(en.coef_)
```

```
[0. 0. 0. 0.]
```

In [75]:
```
print(en.intercept_)
```

```
0.0
```

In [76]:
```
print(en.predict(x_test))
```

```
[0. 0. 0. 0. 0. 0.]
```

In [77]:
```
print(en.score(x_test,y_test))
```

```
1.0
```

In [78]:
```
from sklearn import metrics
```

In [79]:
```
print("Mean Absolute Error",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error 0.0
```

In [80]:
```
print("Mean squared Error",metrics.mean_squared_error(y_test,prediction))
```

```
Mean squared Error 0.0
```

In [81]:
```
import pickle
```

In [82]:
```
filename="prediction"
pickle.dump(lr,open(filename,'wb'))
```

In [83]:
```
import pandas as pd
import pickle
```

In [87]:
```
filename='prediction'
model=pickle.load(open(filename,'rb'))
```

In [91]:
```
real=[[10,20,30,40],[40,55,66,88]]
result=model.predict(real)
```

In [92]:
```
result
```

Out[92]: array([0., 0.])

In [ ]: