Data Cleaning

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

In [2]:
```
df = pd.read_csv(r"C:\Users\user\Downloads\19_nuclear_explosions.csv")
df
```

Out[2]:

	WEAPON SOURCE COUNTRY	WEAPON DEPLOYMENT LOCATION	Data.Source	Location.Cordinates.Latitude	Location.Cordinates.Lo
0	USA	Alamogordo	DOE	32.54	
1	USA	Hiroshima	DOE	34.23	
2	USA	Nagasaki	DOE	32.45	
3	USA	Bikini	DOE	11.35	
4	USA	Bikini	DOE	11.35	
...	...	...	...	...	
2041	CHINA	Lop Nor	HFS	41.69	
2042	INDIA	Pokhran	HFS	27.07	
2043	INDIA	Pokhran	NRD	27.07	
2044	PAKIST	Chagai	HFS	28.90	
2045	PAKIST	Kharan	HFS	28.49	

2046 rows × 16 columns

In [3]:
```
# to display info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2046 entries, 0 to 2045
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   WEAPON SOURCE COUNTRY           2046 non-null   object 
 1   WEAPON DEPLOYMENT LOCATION      2046 non-null   object 
 2   Data.Source                     2046 non-null   object 
 3   Location.Cordinates.Latitude    2046 non-null   float64
 4   Location.Cordinates.Longitude   2046 non-null   float64
 5   Data.Magnitude.Body             2046 non-null   float64
 6   Data.Magnitude.Surface          2046 non-null   float64
 7   Location.Cordinates.Depth       2046 non-null   float64
 8   Data.Yeild.Lower                2046 non-null   float64
 9   Data.Yeild.Upper                2046 non-null   float64
 10  Data.Purpose                    2046 non-null   object 
 11  Data.Name                       2046 non-null   object 
 12  Data.Type                       2046 non-null   object 
 13  Date.Day                        2046 non-null   int64  
 14  Date.Month                      2046 non-null   int64  
 15  Date.Year                       2046 non-null   int64  
dtypes: float64(7), int64(3), object(6)
memory usage: 255.9+ KB
```

In [4]:
```
# t display summerize the data
df.describe()
```

Out[4]:

	Location.Cordinates.Latitude	Location.Cordinates.Longitude	Data.Magnitude.Body	Data.Mag
count	2046.000000	2046.000000	2046.000000	
mean	35.462429	-36.015037	2.145406	
std	23.352702	100.829355	2.625453	
min	-49.500000	-169.320000	0.000000	
25%	37.000000	-116.051500	0.000000	
50%	37.100000	-116.000000	0.000000	
75%	49.870000	78.000000	5.100000	
max	75.100000	179.220000	7.400000	

In [5]:
```
# to display columes
df.columns
```

Out[5]:
```
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')
```

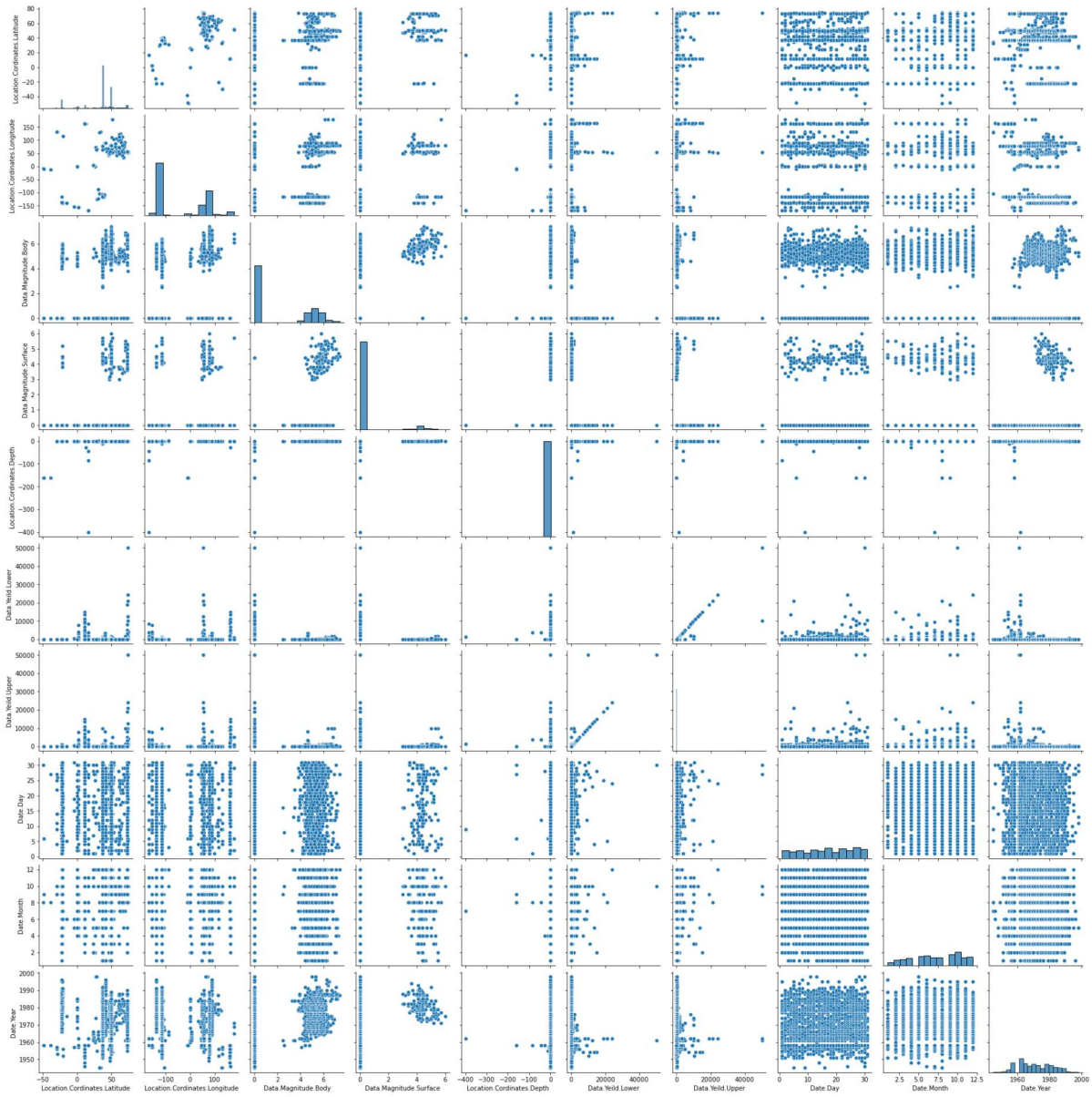
In [6]:
```
df.isna().sum()
```

Out[6]:
```
WEAPON SOURCE COUNTRY           0
WEAPON DEPLOYMENT LOCATION      0
Data.Source                     0
Location.Cordinates.Latitude    0
Location.Cordinates.Longitude   0
Data.Magnitude.Body             0
Data.Magnitude.Surface          0
Location.Cordinates.Depth       0
Data.Yeild.Lower                0
Data.Yeild.Upper                0
Data.Purpose                    0
Data.Name                       0
Data.Type                       0
Date.Day                        0
Date.Month                      0
Date.Year                       0
dtype: int64
```

EDA and visualization

In [7]:
```
sns.pairplot(df)
```

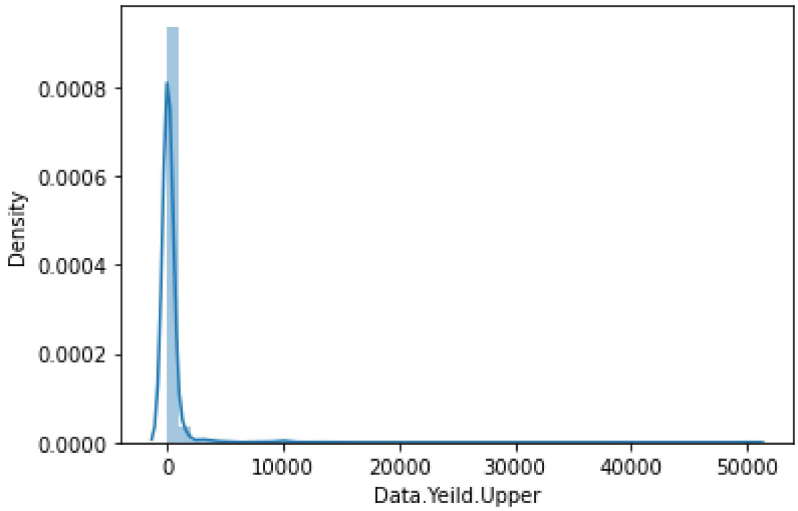
Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x1d57b3fc790>
```



In [8]:
```
# to display distribution graph for price column
sns.distplot(df['Data.Yeild.Upper'])
```

```
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)
```

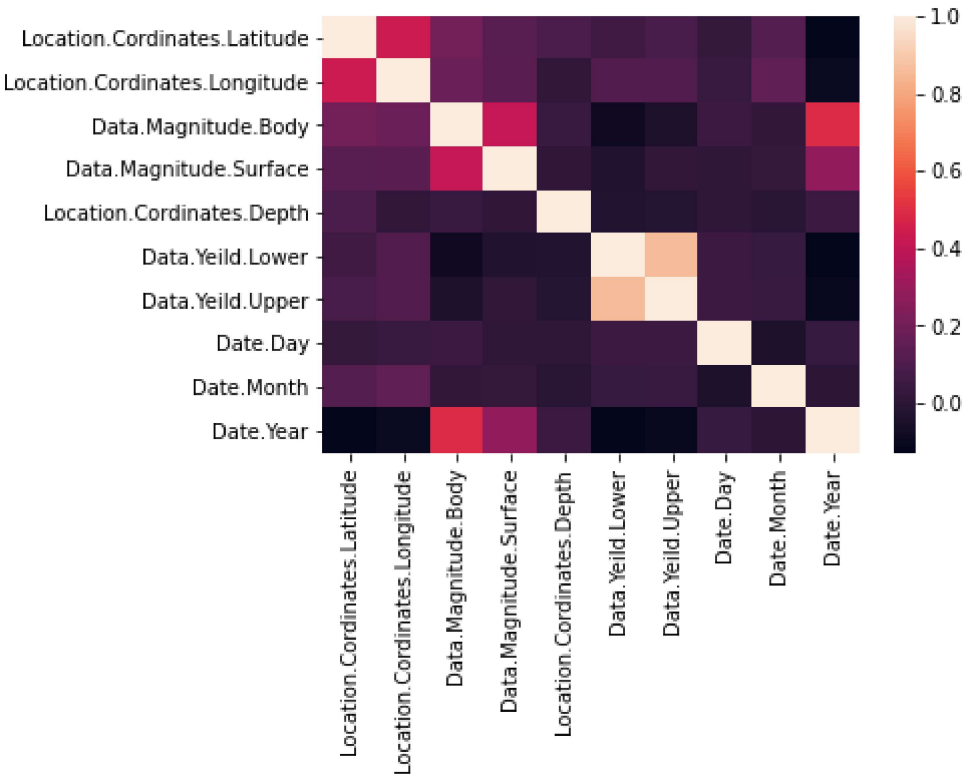
Out[8]: <AxesSubplot:xlabel='Data.Yeild.Upper', ylabel='Density'>



In [9]:
```
df1 = df[['Location.Cordinates.Latitude', 'Location.Cordinates.Longitude',
       'Data.Magnitude.Body', 'Data.Magnitude.Surface',
       'Location.Cordinates.Depth', 'Data.Yeild.Lower', 'Data.Yeild.Upper','Dat
       'Date.Year']]
```

In [10]:
```
# correlation map to find relationship
sns.heatmap(df1.corr())
```

Out[10]: <AxesSubplot:>



In [42]:
```
# Assign x and y for linear regression
x = df1[['Location.Cordinates.Latitude', 'Location.Cordinates.Longitude',
       'Data.Magnitude.Body', 'Data.Magnitude.Surface',
       'Location.Cordinates.Depth', 'Data.Yeild.Upper','Date.Day', 'Date.Month'
       'Date.Year']]
y = df1['Data.Yeild.Lower']
```

In [43]:
```
# to split dataset into training data and test data

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 [44]:
```
#Linear Regression

from sklearn.linear_model import LinearRegression

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

Out[44]: LinearRegression()

In [45]:
```
# intercept is value of c
print(lr.intercept_)
```

```
6862.825310131727
```

In [46]:
```
# co-efficient value of m
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

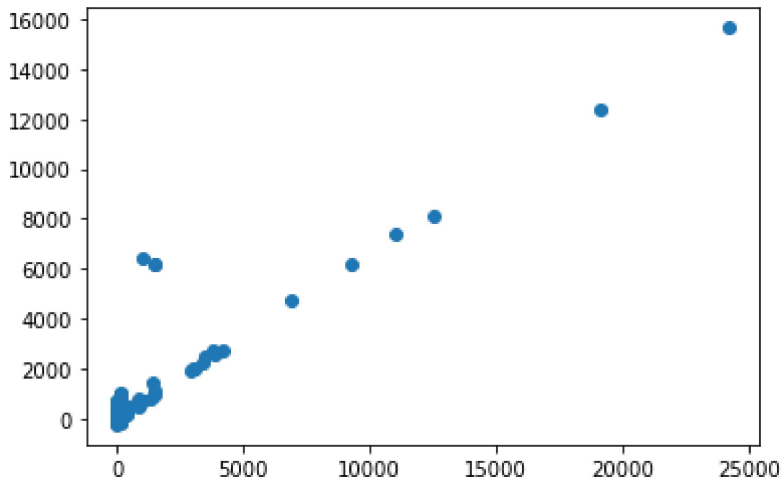
Out[46]:

	Co-efficient
Location.Cordinates.Latitude	-1.780656
Location.Cordinates.Longitude	0.501516
Data.Magnitude.Body	-12.380438
Data.Magnitude.Surface	-30.438619
Location.Cordinates.Depth	-1.397919
Data.Yeild.Upper	0.646023
Date.Day	1.461566
Date.Month	-3.268807
Date.Year	-3.427763

In [47]:
```
#predict the graph in linear regression graph

prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[47]:
```
<matplotlib.collections.PathCollection at 0x1d50d604be0>
```



In [40]:
```
#Accuracy of linear regression

print(lr.score(x_test,y_test))
```

```
-5.580090827428987
```

In [41]:
```
lr.score(x_train,y_train)
```

Out[41]:
```
0.9121373473566865
```

In [19]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [20]:
```
rr = Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

Out[20]:
```
-0.038457062962755195
```

In [21]:
```
rr.score(x_train,y_train)
```

Out[21]:
```
0.7763888210270125
```

In [22]:
```
lr = Lasso(alpha=10)
lr.fit(x_train,y_train)
lr.score(x_test,y_test)
```

Out[22]:
```
-0.041011452935183046
```

In [23]:
```
lr.score(x_train,y_train)
```

Out[23]:
```
0.7763661886777417
```

Elastic

In [24]:
```
from sklearn.linear_model import ElasticNet
es = ElasticNet()
es.fit(x_train,y_train)
```

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

In [25]:
```
print(es.coef_)
```

```
[ -1.1522954    0.62151724 -16.34227328 -22.71308096  -0.83275778
   0.71385896   1.04072888  -3.10190961  -3.40028959]
```

In [26]:
```
print(es.intercept_)
```

```
6807.562094047892
```

In [27]:
```
print(es.score(x_test,y_test))
```

```
-0.041580491242562134
```

Evaluation Model

In [28]:
```
from sklearn import metrics
```

In [29]:
```
print("Mean absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean absolute Error: 147.21837929371645
```

In [30]:
```
print("Mean squared Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean squared Error: 329906.23540453316
```

In [31]:
```
print("Root Mean squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prec
```

```
Root Mean squared Error: 574.3746472508454
```

model saving

In [32]:
```
import pickle # pickle is used to model saving
```

In [33]:
```
filename ="nuclear_explosions prediction"
pickle.dump(lr,open(filename,'wb'))
```

In [48]:
```
df.head(10)
```

Out[48]:

ody	Data.Magnitude.Surface	Location.Cordinates.Depth	Data.Yeild.Lower	Data.Yeild.Upper	Data.Pur
0.0	0.0	-0.10	21.0	21.0	
0.0	0.0	-0.60	15.0	15.0	Cc
0.0	0.0	-0.60	21.0	21.0	Cc
0.0	0.0	-0.20	21.0	21.0	
0.0	0.0	0.03	21.0	21.0	
0.0	0.0	-0.08	37.0	37.0	
0.0	0.0	-0.08	49.0	49.0	
0.0	0.0	-0.08	18.0	18.0	
0.0	0.0	0.00	22.0	22.0	
0.0	0.0	-0.35	1.0	1.0	

In [49]:
```
df.tail(10)
```

Out[49]:

ody	Data.Magnitude.Surface	Location.Cordinates.Depth	Data.Yeild.Lower	Data.Yeild.Upper	Data.Pur
0.0	0.0	0.0	0.0	60.0	
0.0	0.0	0.0	0.0	40.0	
0.0	0.0	0.0	0.0	30.0	
0.0	0.0	0.0	0.0	120.0	
6.3	0.0	0.0	30.0	120.0	
5.3	0.0	0.0	3.0	12.0	
5.3	0.0	0.0	0.0	20.0	
0.0	0.0	0.0	0.0	1.0	
0.0	0.0	0.0	0.0	35.0	
5.0	0.0	0.0	0.0	18.0	

In [ ]: