D16

In [1]:

```
import pandas as pd
import numpy as np
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.Lc
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 [4]:
```
df.head(10)
```

Out[4]:

	WEAPON SOURCE COUNTRY	WEAPON DEPLOYMENT LOCATION	Data.Source	Location.Cordinates.Latitude	Location.Cordinates.Longi
0	USA	Alamogordo	DOE	32.54	-10
1	USA	Hiroshima	DOE	34.23	13
2	USA	Nagasaki	DOE	32.45	12
3	USA	Bikini	DOE	11.35	16
4	USA	Bikini	DOE	11.35	16
5	USA	Enewetak	DOE	11.30	16
6	USA	Enewetak	DOE	11.30	16
7	USA	Enewetak	DOE	11.30	16
8	USSR	Semi Kazakh	DOE	48.00	7
9	USA	Nts	DOE	37.00	-11

In [5]:
```
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 [6]:
```
df.describe()
```

Out[6]:

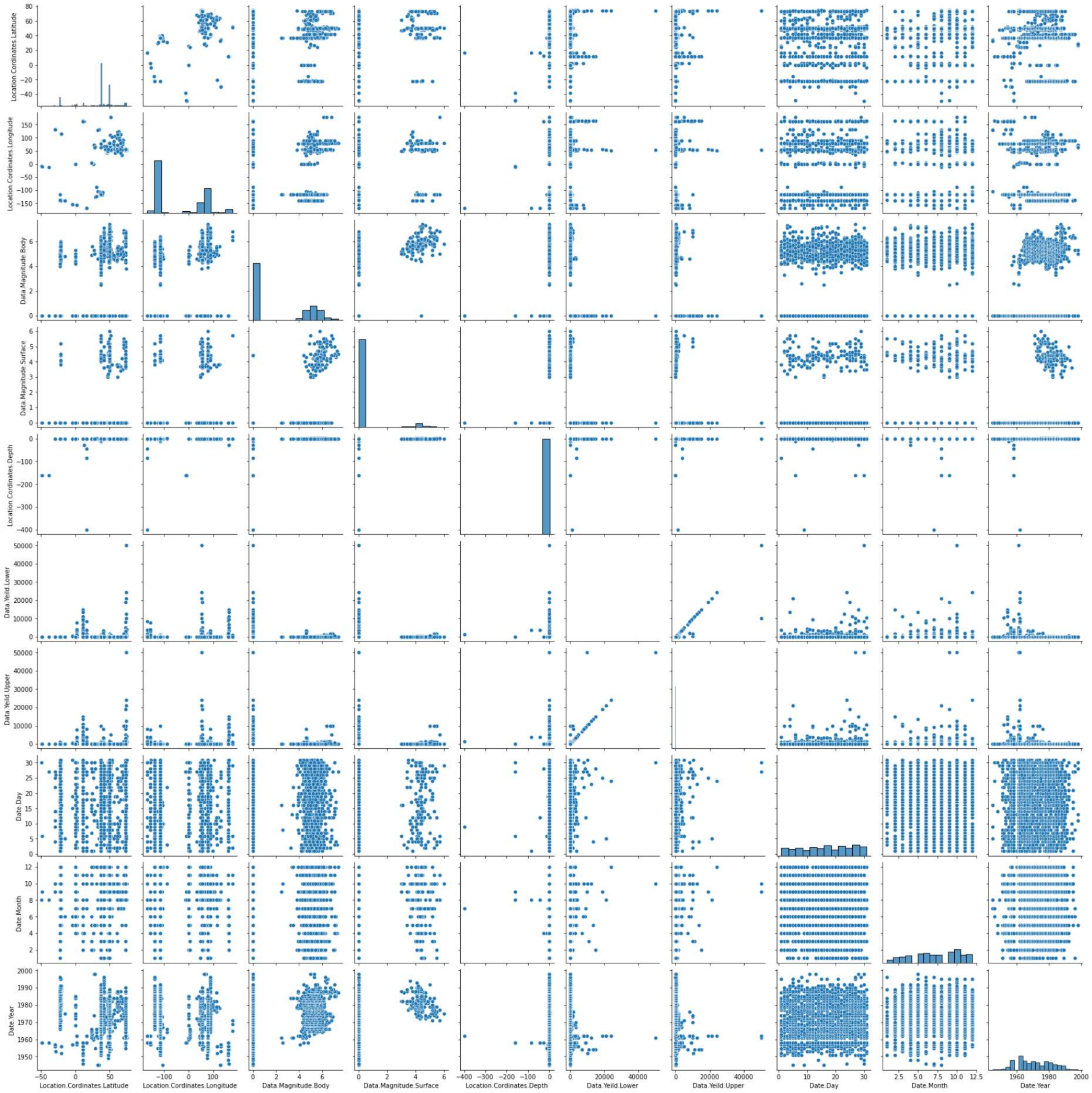
	Location.Cordinates.Latitude	Location.Cordinates.Longitude	Data.Magnitude.Body	Data.Mag
count	2046.000000	2046.000000	2046.000000	
mean	35.462429	-36.015037	2.145406	
std	23.352702	100.829355	2.625453	
min	-49.500000	-169.320000	0.000000	
25%	37.000000	-116.051500	0.000000	
50%	37.100000	-116.000000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

In [7]:
```
df.columns
```

Out[7]:
```
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 [8]:
```
sns.pairplot(df)
```

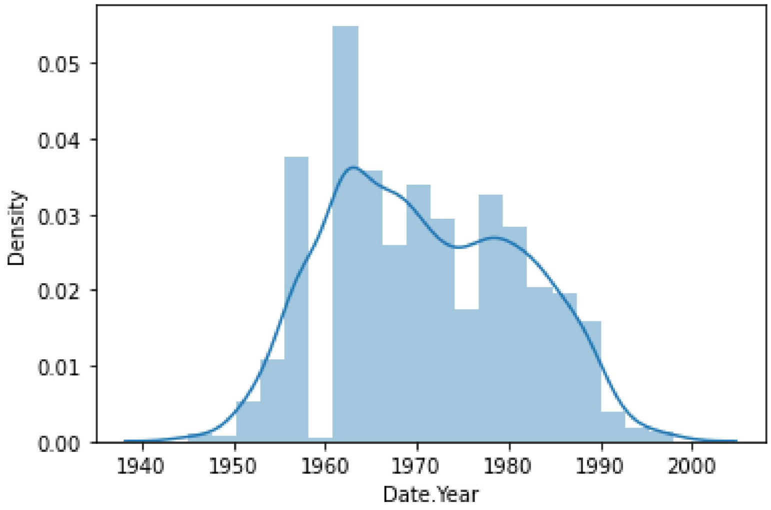
Out[8]:
```
<seaborn.axisgrid.PairGrid at 0x2436244e640>
```



In [10]:
```
sns.distplot(df['Date.Year'])
```

```
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)
```

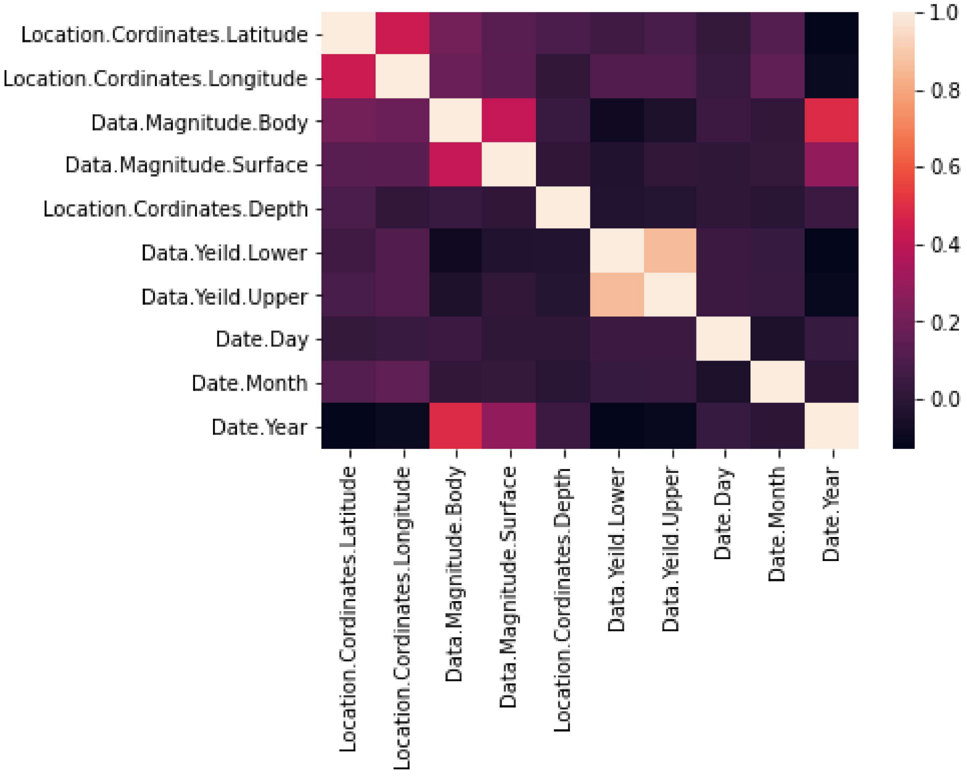
Out[10]:
```
<AxesSubplot:xlabel='Date.Year', ylabel='Density'>
```



In [11]:
```
df1=df[['Location.Cordinates.Latitude', 'Location.Cordinates.Longitude',
       'Data.Magnitude.Body', 'Data.Magnitude.Surface','Location.Cordinates.De
       'Data.Yeild.Upper','Date.Day', 'Date.Month','Date.Year']]
```

In [12]:
```
sns.heatmap(df1.corr())
```

Out[12]:
```
<AxesSubplot:>
```



In [13]:
```
x=df1[['Location.Cordinates.Latitude', 'Location.Cordinates.Longitude',
       'Data.Magnitude.Body', 'Data.Magnitude.Surface','Location.Cordinates.De
       'Data.Yeild.Upper','Date.Day', 'Date.Month']]
y=df1['Date.Year']
```

In [14]:
```
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 [15]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[15]:
```
LinearRegression()
```

In [16]:
```
print(lr.intercept_)
```

```
1967.653018546527
```

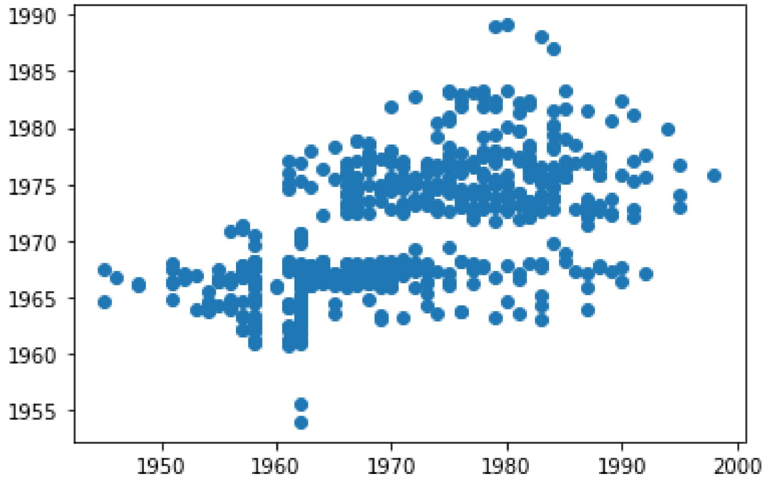
In [17]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[17]:

	Co-efficient
Location.Cordinates.Latitude	-0.091582
Location.Cordinates.Longitude	-0.010558
Data.Magnitude.Body	1.932421
Data.Magnitude.Surface	0.957819
Location.Cordinates.Depth	0.041197
Data.Yeild.Lower	-0.000271
Data.Yeild.Upper	-0.000103
Date.Day	0.029454
Date.Month	0.182038

In [18]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[18]:
```
<matplotlib.collections.PathCollection at 0x243718ee520>
```



In [19]:
```
print(lr.score(x_test,y_test))
```

```
0.3472606195002841
```

In [20]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [21]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[21]:
```
Ridge(alpha=10)
```

In [22]:
```
rr.score(x_test,y_test)
```

Out[22]:
```
0.3472047118567615
```

In [23]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```
Out[23]:
```
Lasso(alpha=10)
```

In [24]:
```
la.score(x_test,y_test)
```
Out[24]:
```
0.16353714885324178
```

In [25]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```
Out[25]:
```
ElasticNet()
```

In [26]:
```
print(en.coef_)
```
```
[-8.55972104e-02 -9.34348943e-03  1.79558295e+00  5.48591959e-01
  3.82436434e-02 -3.43338055e-04 -6.52431655e-05  2.49480195e-02
  1.21896798e-01]
```

In [27]:
```
print(en.intercept_)
```
```
1968.452512060371
```

In [28]:
```
print(en.predict(x_train))
```
```
[1965.08286525 1970.99518333 1965.03989087 ... 1978.00406435 1971.01375701
 1966.98028248]
```

In [29]:
```
print(en.score(x_train,y_train))
```
```
0.312630759683329
```

In [30]:
```
from sklearn import metrics
```

In [31]:
```
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
```
```
Mean Absolytre Error: 6.546865470822468
```

In [32]:
```
print("Mean Square Error:",metrics.mean_squared_error(y_test,prediction))
```
```
Mean Square Error: 67.84926289562725
```

In [33]:
```
print("Root Mean Square Error:",np.sqrt(metrics.mean_absolute_error(y_test,pre
```
```
Root Mean Square Error: 2.558684324183518
```

In [34]:
```
import pickle
```

In [35]:

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

In [ ]: