PES UNIVERSITY
ಪಿಇಎಸ್ ವಿಶ್ವವಿದ್ಯಾಲಯ

ANALYSIS OF OIL PIPELINE ACCIDENTS

BY:

NAME and SRN:

- Nitish S(PES2201800368)
- Sandeep Bhat(PES2201800632)
- Supreet Ronad(PES2201800705)

SECTION: ‘A’

ABSTRACT

Oil is by far the most commonly spilled substance . Since 1986 pipeline accidents have spilled an average of 76,000 barrels per year or more than 3 million gallons. This is equivalent to 200 barrels every day. Oil pipeline accidents not only leads to the wastage of crude oil but also has adverse effects on the environment by contaminating the water and land, leading to water pollution and soil pollution. It also affects the economy of the country as the repairs caused by the accident is huge. Oil leakages in the ocean toxicate aquatic animals and plants thereby affecting the aquatic ecosystem. Oil being a fossil fuel takes millions of years to form , should be transported carefully with adequate measures to prevent leakage. Data collected from various sources helps us to infer on the causes related to oil pipeline accidents.

OIL PIPELINE ACCIDENTS

1. DESCRIPTION OF THE DATASET:

- *Name of the dataset:* **Oil Pipeline Accidents**
- *Number of rows:* **2796**
- *Number of columns:* **30**
- *NUMBER OF CATEGORICAL DATA COLUMNS:* **10** (Accident year, Pipeline location, Pipeline type, Liquid type, Liquid subtype, Cause category, Cause subcategory, Liquid ignition, Liquid explosion, Pipeline shutdown)
- *NUMBER OF NUMERICAL COLUMNS:* **13** (Unintentional release(barrels), Intentional release(barrels), Liquid recovery(barrels), Net loss(barrels), Environmental remediation costs, all costs, Public evacuation, Property damage costs, Lost commodity costs, public/private property damage costs, Emergency response costs, other costs, Public evacuation)
- *Percentage of missing values or NaN:* **Approx 8%**

OIL PIPELINE ACCIDENTS

November 22, 2019

```
In [5]: df['Liquid Name'].head(20)
```

```
Out[5]: 0                  NaN
        1                  NaN
        2               ETHANE
        3                  NaN
        4                  NaN
        5                  NaN
        6                  NaN
        7                  NaN
        8                  NaN
        9                  NaN
        10      NORMAL BUTANE
        11                 NaN
        12                 NaN
        13                 NaN
        14                 NaN
        15                 NaN
        16                 NaN
        17            PROPYLENE
        18                 NaN
        19                 NaN
        Name: Liquid Name, dtype: object
```

BEFORE CLEANING

```
In [25]: df=df.drop(columns='Liquid Name')
         df.to_csv("C:\\Users\\Nitish Srivatsa\\.spyder-py3\\properset1.csv")
```

```
In [26]: df=df.drop(columns='Public Evacuations')
         df.to_csv("C:\\Users\\Nitish Srivatsa\\.spyder-py3\\properset1.csv")
```

DROPPING OF THE COLUMNS WITH TOO MANY NaN VALUES

2) The numerical columns containing the costs needed to restore the damages caused because of the pipeline accidents have been filled with the mean of values of the respective columns.

```
df['Environmental Remediation Costs'].head(20)
```

```
0           0.0
1           0.0
2           0.0
3           0.0
4        2000.0
5     2000000.0
6       70000.0
7           0.0
8       40000.0
9       10000.0
10          NaN
11     308883.0
12       6500.0
13        100.0
14          0.0
15          0.0
16    1901615.0
17          0.0
18      30000.0
19      64645.0
```

BEFORE CLEANING

```
Out[49]: 0                   0.0
         1                   0.0
         2                   0.0
         3                   0.0
         4                2000.0
         5             2000000.0
         6               70000.0
         7                   0.0
         8               40000.0
         9               10000.0
         10     362809.44025834225
         11             308883.0
         12               6500.0
         13                100.0
         14                  0.0
         15                  0.0
         16            1901615.0
         17                  0.0
         18              30000.0
         19              64645.0
         Name: Environmental Remediation Costs, dtype: object
```

NaN replaced by mean

AFTER CLEANING

```
array4=df[df['Environmental Remediation Costs']!=np.nan]['Environmental Remediation Costs']
df['Environmental Remediation Costs']=df['Environmental Remediation Costs'].replace(np.nan,array4.mean())
```

PROCEDURE ADOPTED FOR REPLACING NaN VALUES WITH MEAN

Similar procedure is followed for the other columns containing the costs and the NaN values are replaced by mean.

3. Normalization and Standardization:

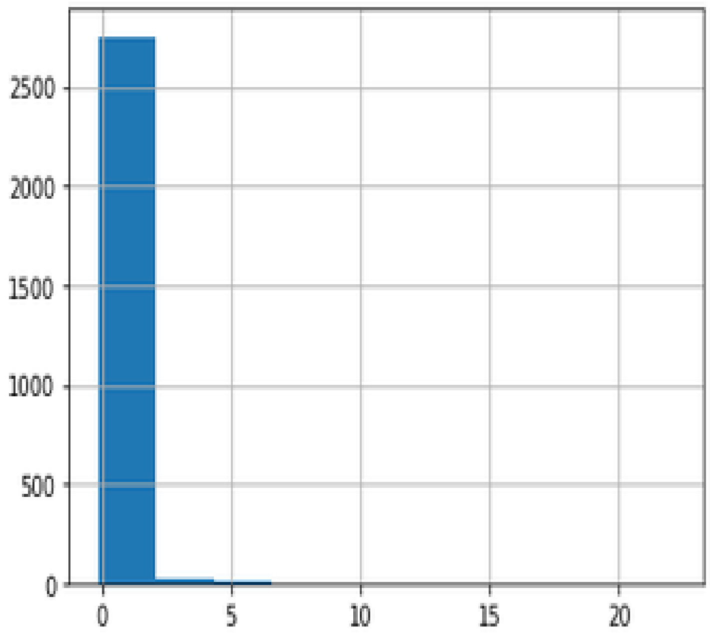
Normalization is a type of process wherein data within a database is reorganized in such a way so that users can properly utilize that database for further queries and analysis.

GOALS:

- To get rid of any duplicate data that might appear within the data set.
- To eliminate any redundancies that may occur as Redundancies can adversely affect analysis of data since they are values which aren’t exactly needed.
- To get rid of a number of anomalies that can make analysis of the data more complicated. Some of those anomalies can crop up from deleting data, inserting more information, or updating existing information. Once those errors are worked out and removed from the system, further benefits can be gained through other uses of the data and data analytics.

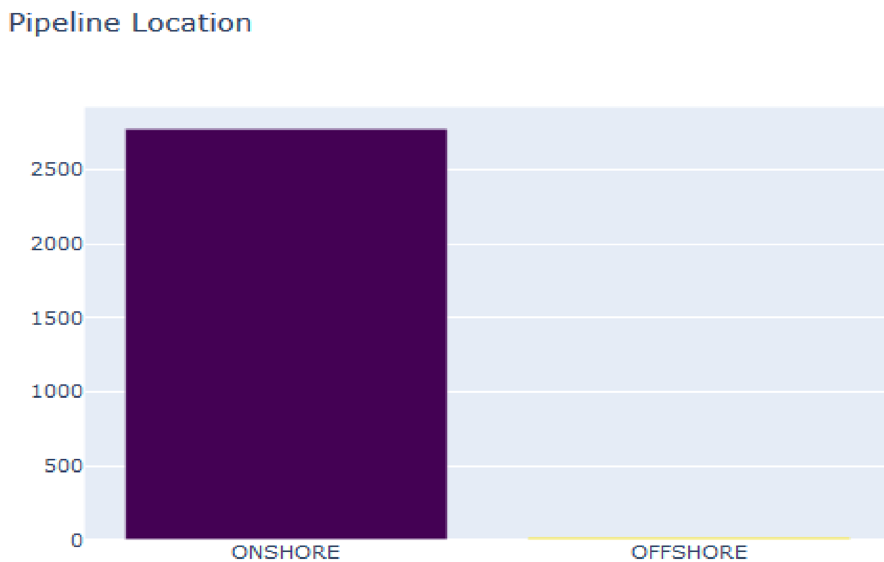
Standardized data is essential for accurate data analysis; it's easier to draw clear conclusions about your current data when you have other data to measure it against.

```
In [41]: df=pd.read_csv("C:\\Users\\Nitish Srivatsa\\.spyder-py3\\normalset.csv")

In [42]: df['Unintentional Release (Barrels)'].hist()

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0xf29ac43f88>
```



From the above histogram, we can say that the column 'Unintentional Release' is right skewed after normalizing and plotting the graph.

4. Graph visualization:



PLOT FOR PIPELINE LOCATION Vs NUMBER OF ACCIDENTS

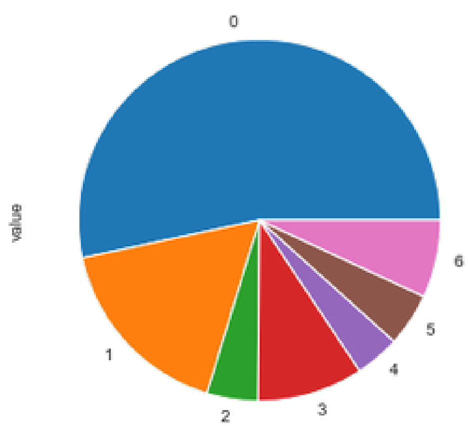
- The above plot shows the number of accidents caused due to the location of the pipeline (ie ONSHORE or OFFSHORE).
- It is clear from the graph that Onshore accidents are more frequent than Offshore. It is mainly related to the material and weld differences as shown in the previous graph.
- Hence we can conclude by saying Onshore establishments need better weld and more maintainence to work smoothly compared to Offshore establishments.

Out[86]:

	type	value
0	equip fail	1243774427
1	corrosion	395325677
2	incorrect operation	106140454
3	natural force damage	220354295
4	excavation damage	93101223
5	others	110824821
6	outside force	161602026

In [87]:
```
pb['value'].plot.pie(figsize=(5,5))
```

Out[87]: <matplotlib.axes._subplots.AxesSubplot at 0x51e6ac4088>



PLOT SHOWING THE AMOUNT SPENT TO RE-ESTABLISH THE PIPELINE Vs THE ACCIDENTS OCCURRED DUE TO VARIOUS CAUSES

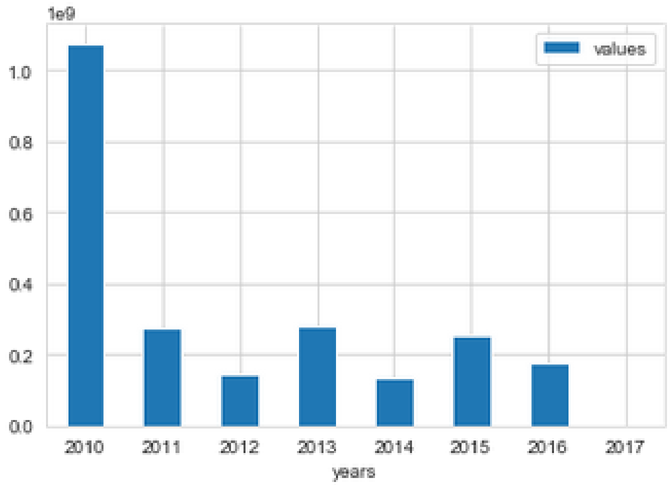
- The above plot depicts the amount spent to re-establish the pipeline due to different accident causes.
- It gives a very effective visualization about the cause which leads to more accidents based on the cost for re-establishment and as a result helps in implementing methods to avoid accidents by such faults to the maximum possible.

Out[49]:

	years	values
0	2010	1075193990
1	2011	273526547
2	2012	145247426
3	2013	278525540
4	2014	131684524
5	2015	253696042
6	2016	173161626
7	2017	87228

In [50]:
```
pb.plot.bar(x='years', y='values', rot=0)
```

Out[50]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x51e67cce48>
```



COSTS Vs
ACCIDENT YEARS

- Though we cannot come up with a generalization about anything through the above plot, one of the noticeable feature is that there are attempts being made from year to year to reduce the number of accidents which is clear from the reduction in the costs for re-establishment in the recent years, suggesting that there are efficient methods being implemented.

5. Hypothesis Testing:

- Null Hypothesis: The proportion of ABOVEGROUND pipeline accidents in the year 2010 is not less than 0.47,
  H_o: p_o**>=0.47**
- Alternate Hypothesis: The proportion of ABOVEGROUND pipeline accidents in the year 2010 is less than 0.47,
  H_1: p_o<**0.47**

In [5]:
```
above_2010=0
for i in range(0,2795):
    if (df['Accident Year'][i]==2010 and df['Pipeline Type'][i]=='ABOVEGROUND'):
        above_2010=above_2010+1
```

In [6]:
```
above_2010
```

Out[6]: 166

In [7]:
```
#population proportion
p=above_2010/350
```

In [8]:
```
p
```

Out[8]: 0.4742857142857143

In [14]:
```
above_2010=0
for i in range(0,500):
    if (df['2015'][i]==2010 and df['ABOVEGROUND'][i]=='ABOVEGROUND'):
        above_2010=above_2010+1
```

In [15]:
```
above_2010
```

Out[15]: 36

In [17]:
```
#sample proportion
p_cap=above_2010/64
```

In [18]:
```
p_cap
```

Out[18]: 0.5625

```
In [76]: #test statistic (z-value)
         z=(p_cap-p)/(((p*(1-p)/500))**0.5)
```

```
In [23]: z
Out[23]: 3.9502903051080547
```

```
In [24]: #function to calculate p value for the z value
         import numpy as np
         import scipy.special as scsp
         def ztop(z):
             return 0.5*(1+scsp.erf(z/np.sqrt(2)))
```

```
In [25]: p_value=ztop(z)
```

```
In [26]: p_value
Out[26]: 0.9999609717704078
```

Assuming α=0.05

We got **p_value > α** and so we do not reject the Null Hypothesis.

Hence, **The proportion of ABOVEGROUND pipeline accidents in the year 2010 is not less than 0.47,**

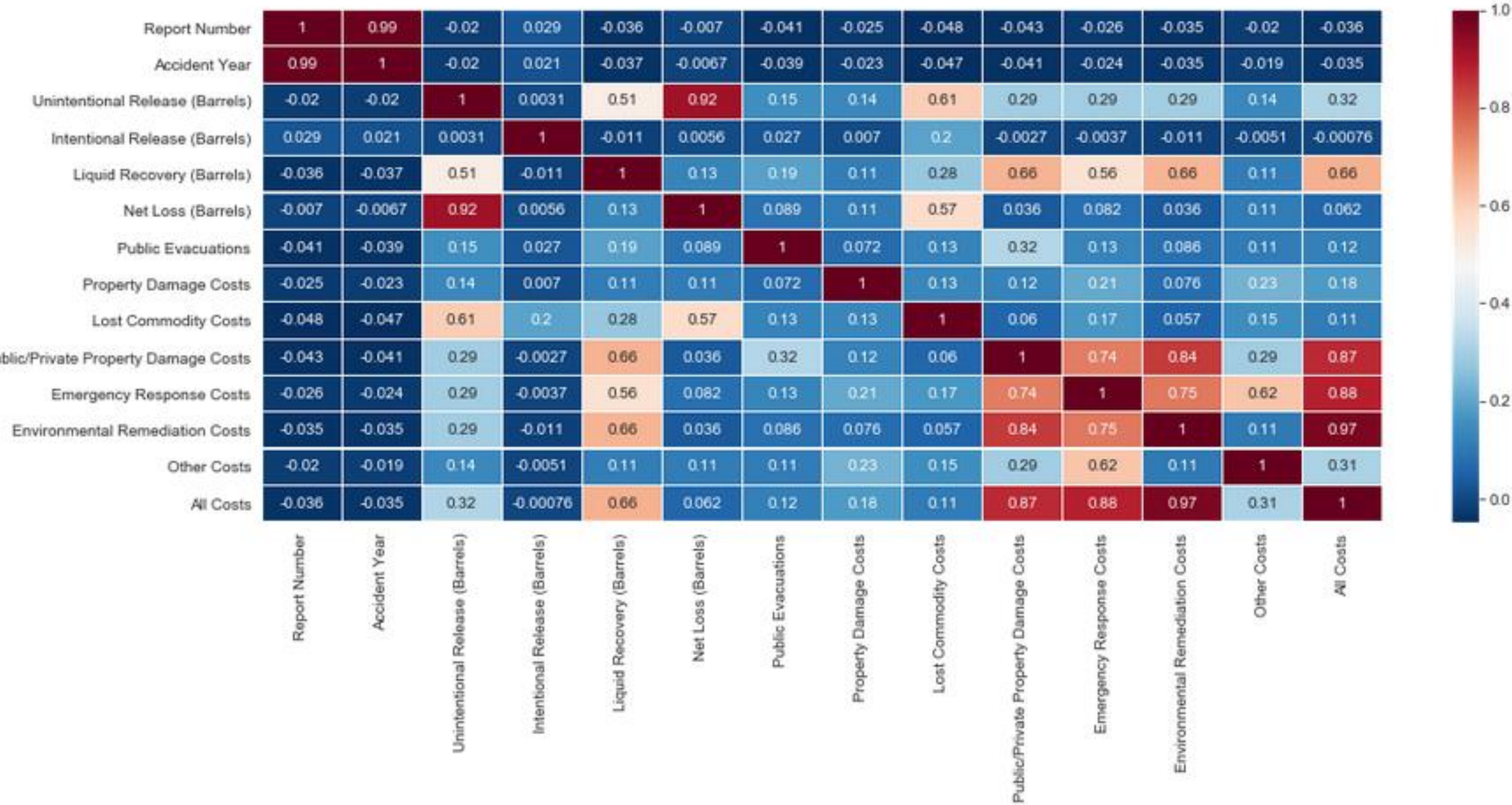
6. Correlation:

Correlation in data analytics is the mutual relationship between two or more variables.

CODE TO CALCULATE CORRELATION COEFFICIENTS

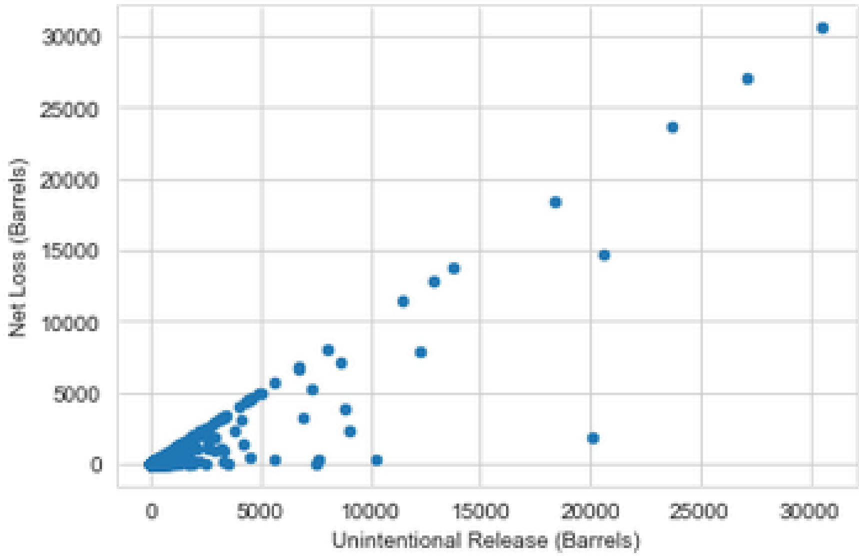
```
In [7]: import matplotlib.pyplot as plt
        import numpy as np
        plt.figure(figsize=(16,6))
        pearsoncorr=df.corr(method='pearson')
        pearsoncorr
        ax=sns.heatmap(pearsoncorr, xticklabels=pearsoncorr.columns,
                    yticklabels=pearsoncorr.columns,
                    cmap='RdBu_r',annot=True,linewidth=0.2)
        bottom,top=ax.get_ylim()
        ax.set_ylim(bottom+0.5,top-0.5)
```

CORRELATION COEFFICIENTS



PLOT BETWEEN UNINTENTIONAL RELEASE Vs NET LOSS

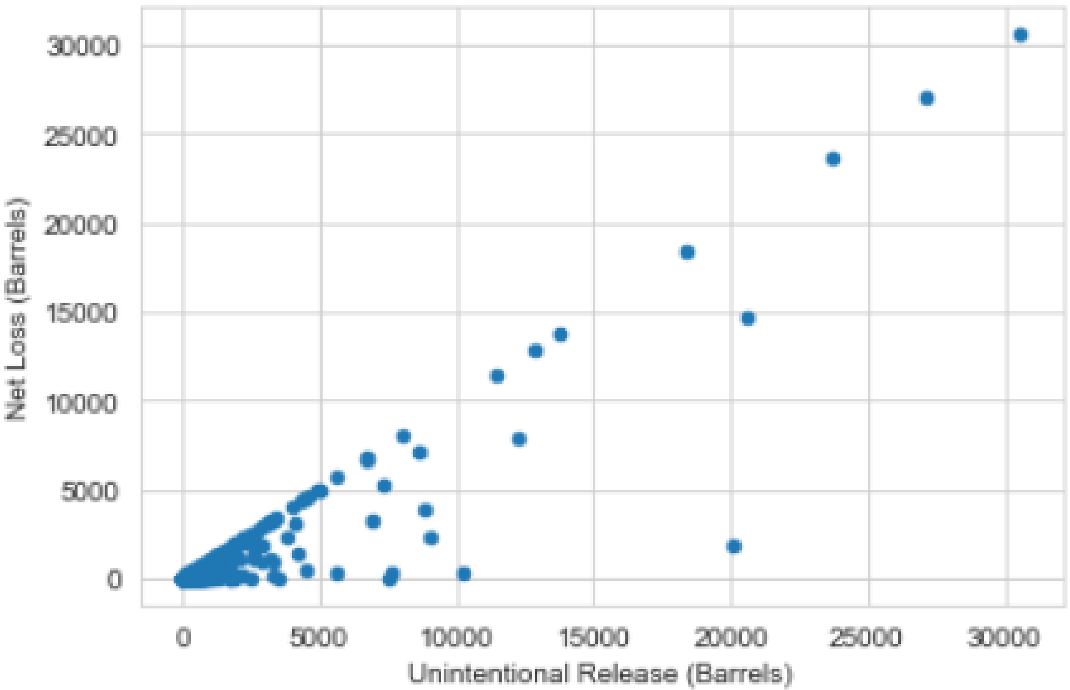
```
In [8]: df.plot.scatter(x='Unintentional Release (Barrels)',y='Net Loss (Barrels)')
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x27f791af88>
```



CORRELATION COEFFICIENT: **r = 0.92**

PLOT BETWEEN UNINTENTIONAL RELEASE Vs INTENTIONAL RELEASE

```
In [9]: df.plot.scatter(x='Unintentional Release (Barrels)',y='Intentional Release (Barrels)')
Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x27f7c28648>
```



CORRELATION COEFFICIENT: **r = 0.0031**

THANK YOU