UNVEILING PATTERNS IN OIL SPILLS OVER TIME

DATA COLLECTION

Import necessary libraries

The occurrence of oil spills represents a significant environmental and economic concern globally. These incidents can have profound and lasting impacts on marine and coastal ecosystems, wildlife, and local communities. Understanding the patterns, trends, and factorscontributing to oil spillage is crucial for effective environmental management and policy- making.

This project focuses on exploring the historical data of oil spill incidents from 1950 onwards. Byconducting exploratory data analysis (EDA), we aim to uncover insights into the frequency, magnitude, and geographical distribution of oil spills over different decades. Through visualizations and statistical summaries, this analysis seeks to identify trends, highlight notableevents, and assess changes in spillage patterns over time.

These datasets are like treasure maps,they are packed with historical data on oil spill,includingthe number of spills,how much oil was spilled,the size of spills.

```
#import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Load the Dataset

```
#load the dataset
number_of_oil_spills = pd.read_csv('/Users/saideepak/Desktop/1-
number-oil-spills.csv')
quantity_of_oil_spills = pd.read_csv('/Users/saideepak/Desktop/2-
quantity-oil-spills.csv')
large_oil_spills_decadal = pd.read_csv('/Users/saideepak/Desktop/3-
large-oil-spills-decadal.csv')
quantity_oil_spills_decadal = pd.read_csv('/Users/saideepak/Desktop/4-
quantity-oil-spills-decadal-average.csv')
```

Data Cleaning and Preprocessing

Data cleaning ensures that the data is accurate, reliable, and consistent, which is crucial for obtaining meaningful insights and making informed decisions.Data cleaning include handlingmissing data, removing duplicates, correcting data types, and dealing with outliers.

initial data exploration

```
#display the first few rows of the dataset
print(number_of_oil_spills.head())

#Display the shape of the dataset
print('Shape of the dataset:',number_of_oil_spills.shape)
```

```
#Display the column names of the dataset
print('Columns in the dataset:',number_of_oil_spills.columns)

#Display the data types of the columns
print('Data typess of the columns:',number_of_oil_spills.dtypes)

#Display the number of missing values in each column
print('Number of missing values in each
column:',number_of_oil_spills.isnull().sum())
```

```
  Entity      Code  Year  Large oil spills (>700 tonnes)  \
0  World  OWID_WRL  1970                              29
1  World  OWID_WRL  1971                              14
2  World  OWID_WRL  1972                              27
3  World  OWID_WRL  1973                              31
4  World  OWID_WRL  1974                              27

   Medium oil spills (7-700 tonnes)
0                                 7
1                                18
2                                48
3                                28
4                                90
Shape of the dataset: (53, 5)
Columns in the dataset: Index(['Entity', 'Code', 'Year', 'Large oil
spills (>700 tonnes)',
       'Medium oil spills (7-700 tonnes)'],
      dtype='object')
Data typess of the columns: Entity                                object
Code                                object
Year                                 int64
Large oil spills (>700 tonnes)       int64
Medium oil spills (7-700 tonnes)     int64
dtype: object
Number of missing values in each column: Entity
0
Code                                0
Year                                0
Large oil spills (>700 tonnes)      0
Medium oil spills (7-700 tonnes)    0
dtype: int64
```

Data Cleaning

```
#Check for outliers
#For simplicity we will use Z-score method to detect and remove
outliers.
from scipy import stats
z_scores =
```

```
np.abs(stats.zscore(number_of_oil_spills.select_dtypes(include=[np.num
ber])))
number_of_oil_spills = number_of_oil_spills[(z_scores <
3).all(axis=1)]
```

EXPLORATORY DATA ANALYSIS(EDA)

Exploratory Data Analysis (EDA) is a crucial initial step in analyzing any dataset, including oil spillage data from 1970 onwards. It involves summarizing the main characteristics of the data, often using graphical and statistical techniques. EDA helps to uncover patterns, identify outliers, and test assumptions, providing insights that guide further analysis and hypothesis generation. By visualizing data distributions, relationships, and trends, EDA enables researchers to make informed decisions about subsequent modeling or investigation strategies. It's a fundamental process for understanding the underlying story within the data before diving into more advanced analytics or interpretation.

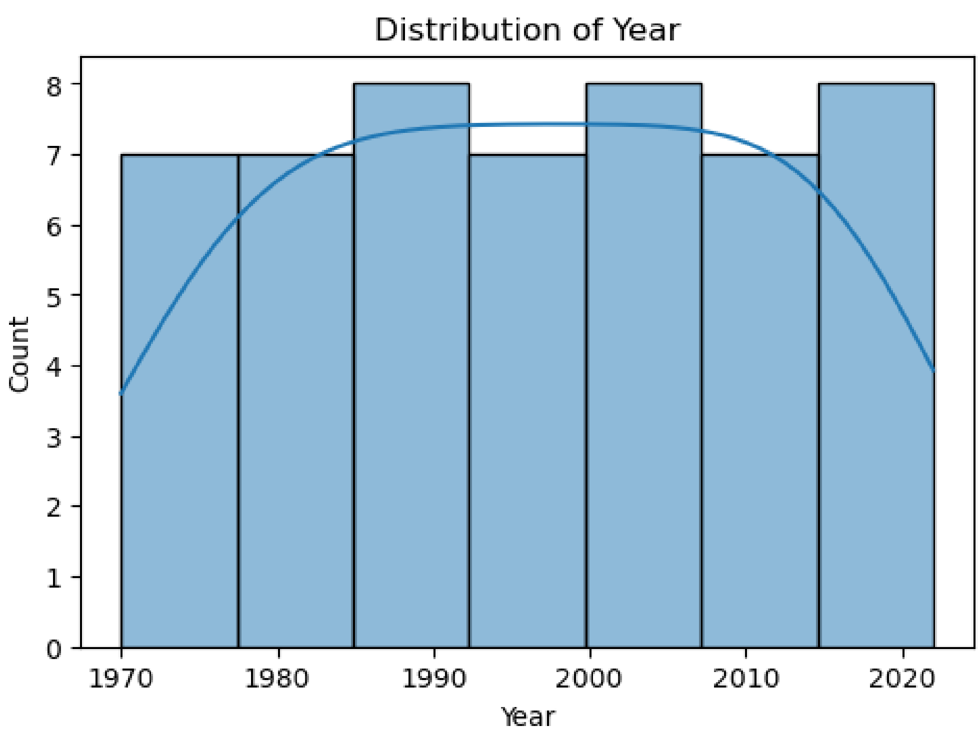
Analyze the Distribution of Data

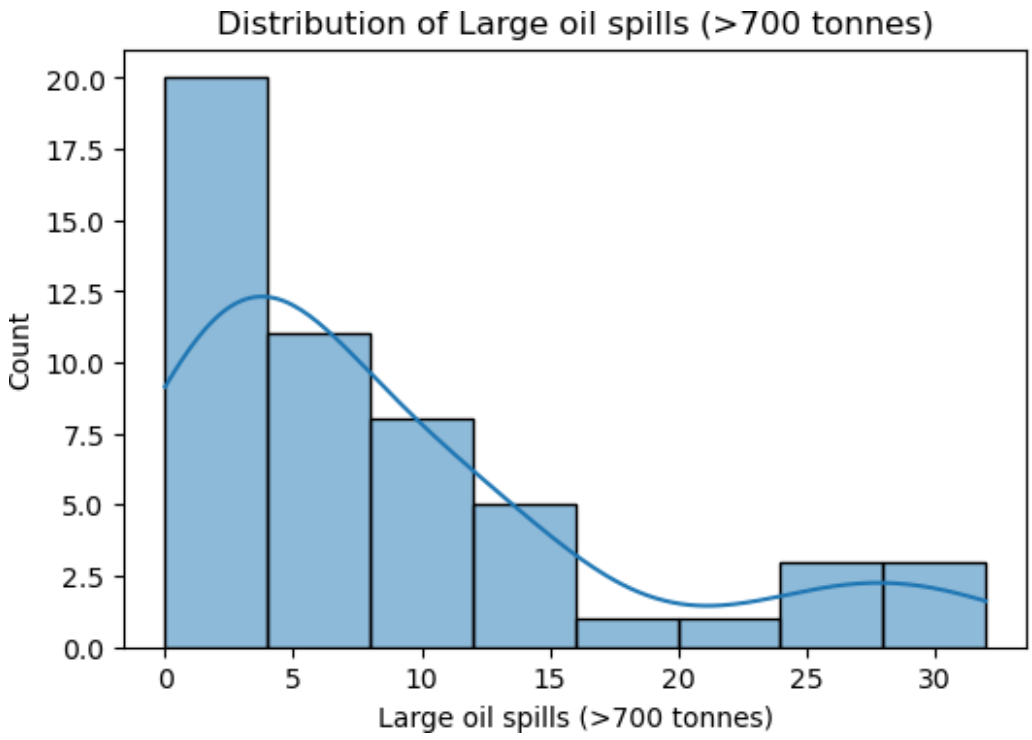
```
#By using descriptive statistics to understand the distribution of
data in each column
print(number_of_oil_spills.describe())
```

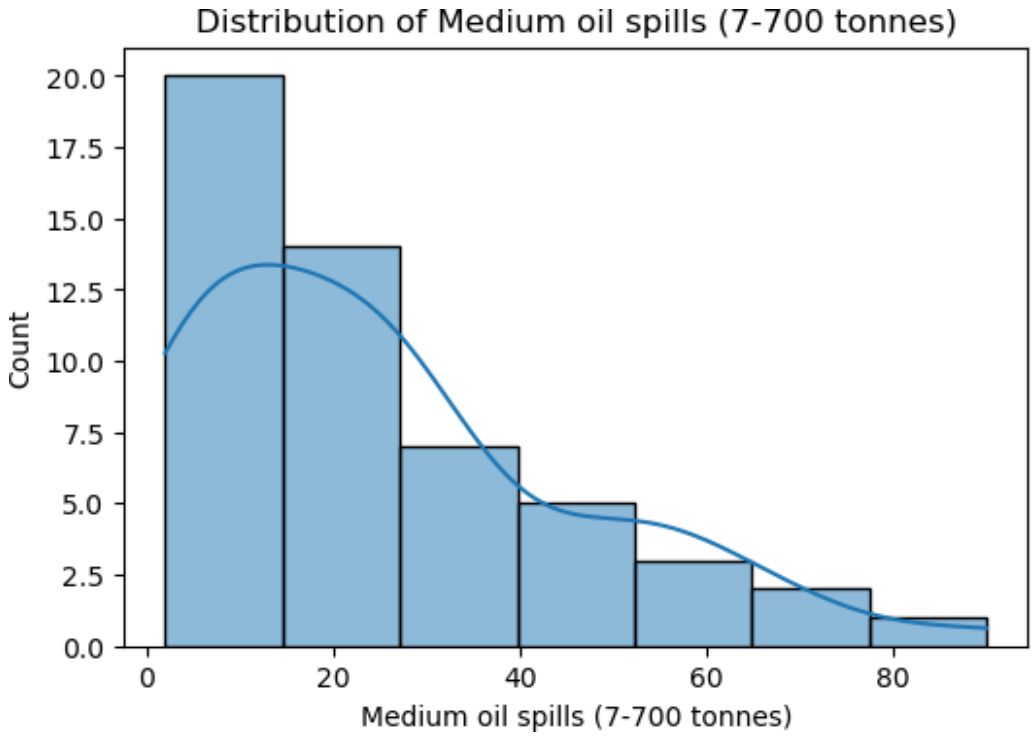
```
              Year  Large oil spills (>700 tonnes)  \
count    52.000000                        52.000000
mean   1996.403846                         8.653846
std      15.308923                         8.746730
min    1970.000000                         0.000000
25%    1983.750000                         3.000000
50%    1996.500000                         5.000000
75%    2009.250000                        11.500000
max    2022.000000                        32.000000

       Medium oil spills (7-700 tonnes)
count                         52.000000
mean                          24.903846
std                           20.764090
min                            2.000000
25%                            7.000000
50%                           20.000000
75%                           31.250000
max                           90.000000
```

```
#Using visualizing techniques to understand the distribution of data
in each column
#Histograms for numerical columns
for column in
number_of_oil_spills.select_dtypes(include=[np.number]).columns:
    plt.figure(figsize=(6,4))
    sns.histplot(number_of_oil_spills[column],kde=True)
```

```
    plt.title(f'Distribution of {column}')
    plt.show

/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



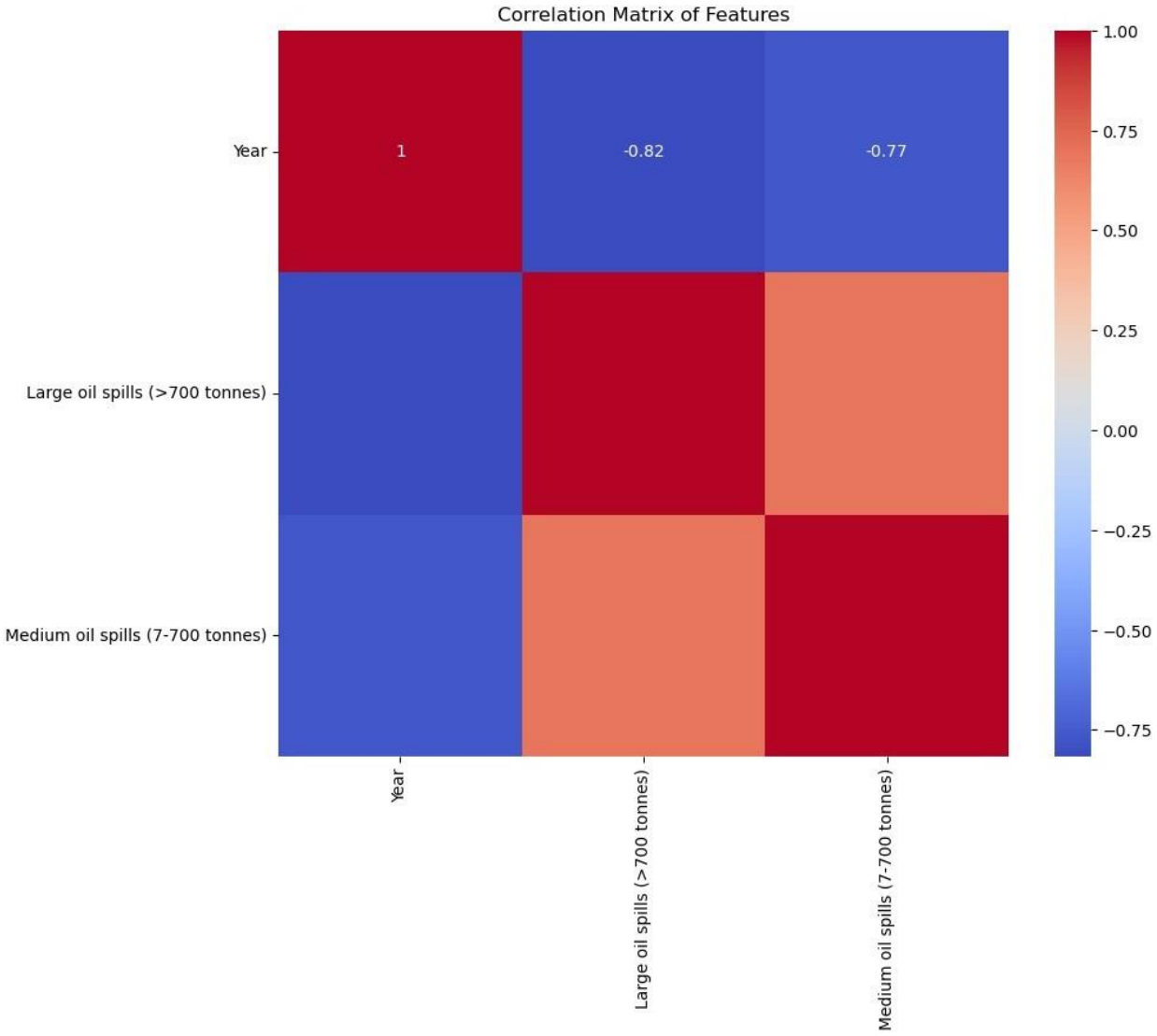




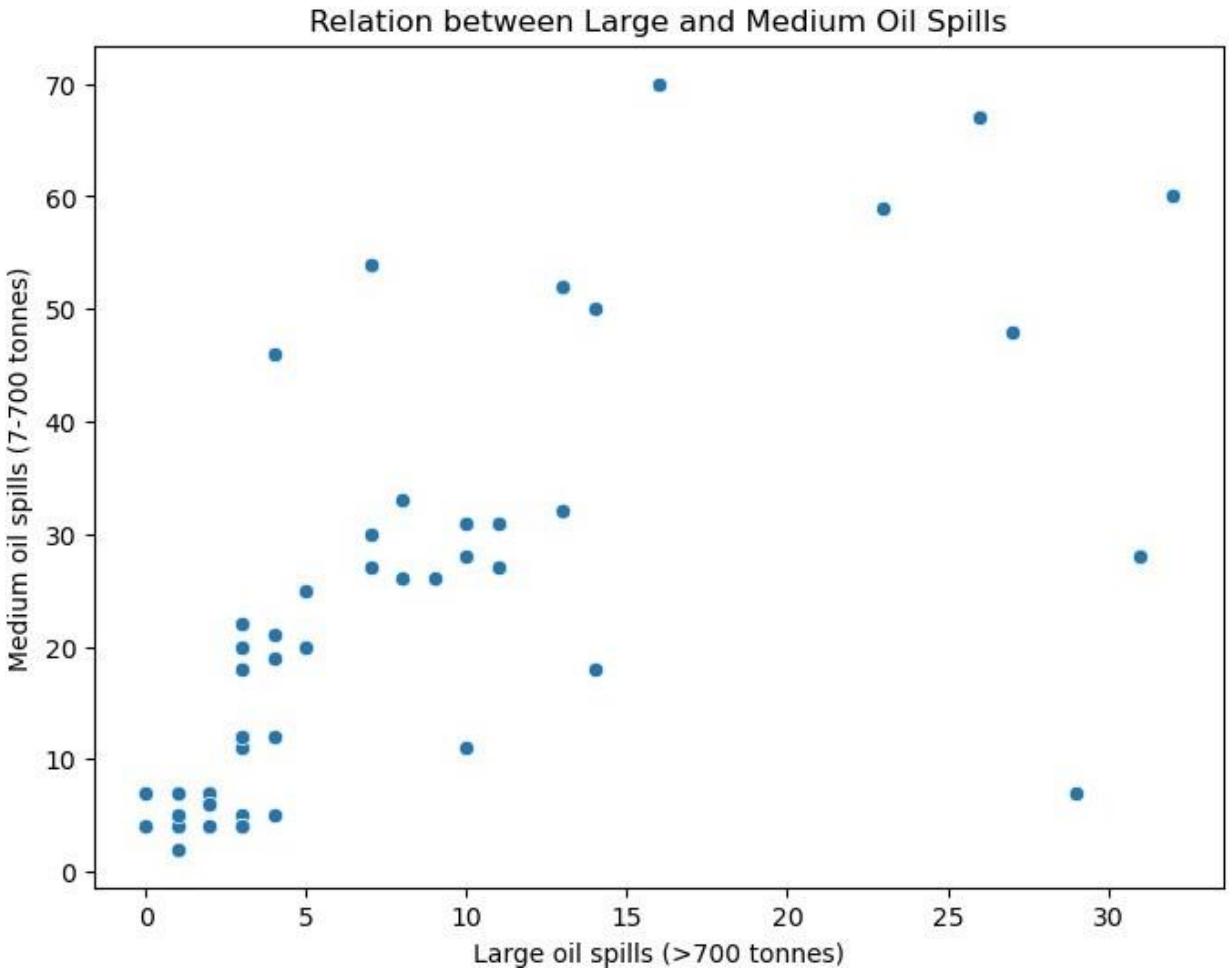
Analyze the relation between different Features

```
#Use correlation analysis to understand the relation between different features
```

```
correlation_matrix =
number_of_oil_spills.select_dtypes(include=[np.number]).corr()
plt.figure(figsize=(10,8))
sns.heatmap(correlation_matrix, annot=True,cmap='coolwarm')
plt.title('Correlation Matrix of Features')
plt.show()
```




```
#Using Scatter plots to visualize the relation between features
plt.figure(figsize=(8,6))
sns.scatterplot(data=number_of_oil_spills,x='Large oil spills (>700
tonnes)',y='Medium oil spills (7-700 tonnes)')
plt.title('Relation between Large and Medium Oil Spills')
plt.show()
```



Feature Engineering

Based on the results we got in EDA,we can create new features.For example,we can create a new feature 'Total Oil Spills" which is sum of 'Large oil spills (>700 tonnes)' and 'Medium oil spills (7-700 tonnes)'.This feature can allow us is in more flexible way of understanding the trends of oil spills.

```
# create a new feature 'Total Oil Spills" which is sum of 'Large oil spills (>700 tonnes)' and 'Medium oil spills (7-700 tonnes)'
number_of_oil_spills['Total oil spills'] = number_of_oil_spills['Large oil spills (>700 tonnes)'] + number_of_oil_spills['Medium oil spills (7-700 tonnes)']

#Display the first few rows of the dataset to check the new feature
print(number_of_oil_spills.head())
```

```
  Entity      Code  Year  Large oil spills (>700 tonnes)  \
0  World  OWID_WRL  1970                              29
1  World  OWID_WRL  1971                              14
2  World  OWID_WRL  1972                              27
```

```
3  World  OWID_WRL  1973                                  31
6  World  OWID_WRL  1976                                  26

   Medium oil spills (7-700 tonnes)  Total oil spills
0                                 7                36
1                                18                32
2                                48                75
3                                28                59
6                                67                93

/var/folders/bc/w1php13n09z2npz3h9jssyqh0000gn/T/
ipykernel_75307/1149025767.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  number_of_oil_spills['Total oil spills'] =
number_of_oil_spills['Large oil spills (>700 tonnes)'] +
number_of_oil_spills['Medium oil spills (7-700 tonnes)']
```

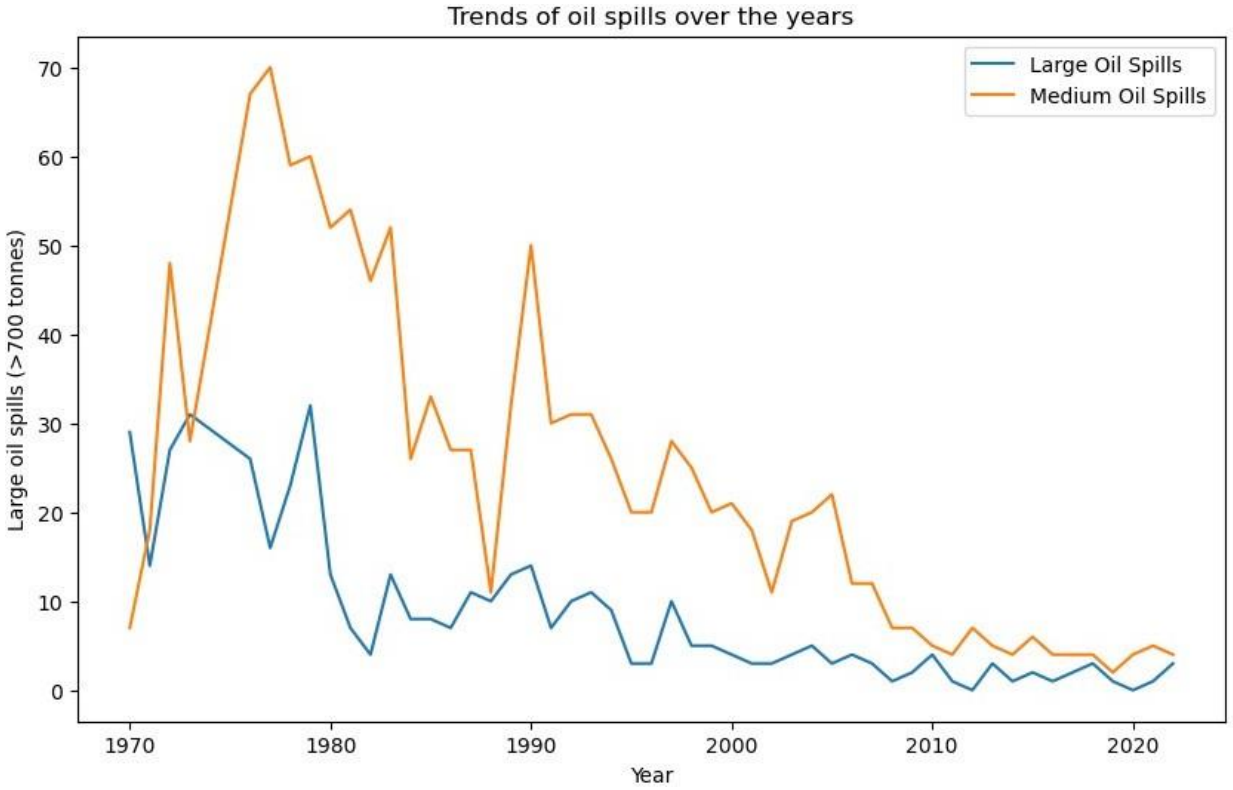
TREND ANALYSIS

Trend analysis is a method used to identify and interpret patterns in data over time.It is the graphical representation using line plots,bar plots and other visualisatiion techniques to visualize these trends.we will look at the trends of oil spills,quanmtity of oil spilled and the size of spills over the years.

```
#Analyze the trends of oil over the years
plt.figure(figsize=(10,6))
sns.lineplot(data=number_of_oil_spills.select_dtypes(include=[np.numbe
r]),x='Year',y='Large oil spills (>700 tonnes)',label='Large Oil
Spills')
sns.lineplot(data=number_of_oil_spills.select_dtypes(include=[np.numbe
r]),x='Year',y='Medium oil spills (7-700 tonnes)',label='Medium Oil
Spills')
plt.title('Trends of oil spills over the years')
plt.legend()
plt.show()
```

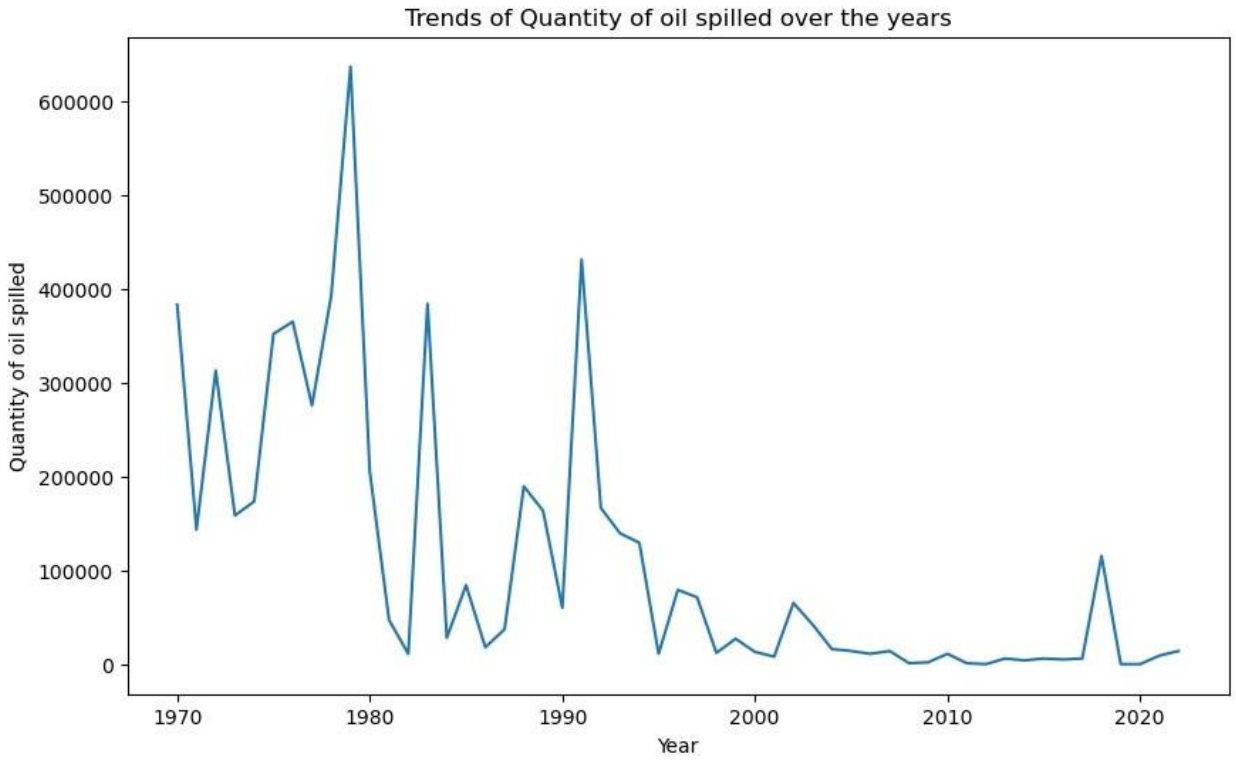
```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
```

```
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
#Analyze the trends of the quantity of oil spilled over the years
plt.figure(figsize=(10,6))
sns.lineplot(data=quantity_of_oil_spills,x='Year',y='Quantity of oil
spilled')
plt.title('Trends of Quantity of oil spilled over the years')
plt.show()

/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
```

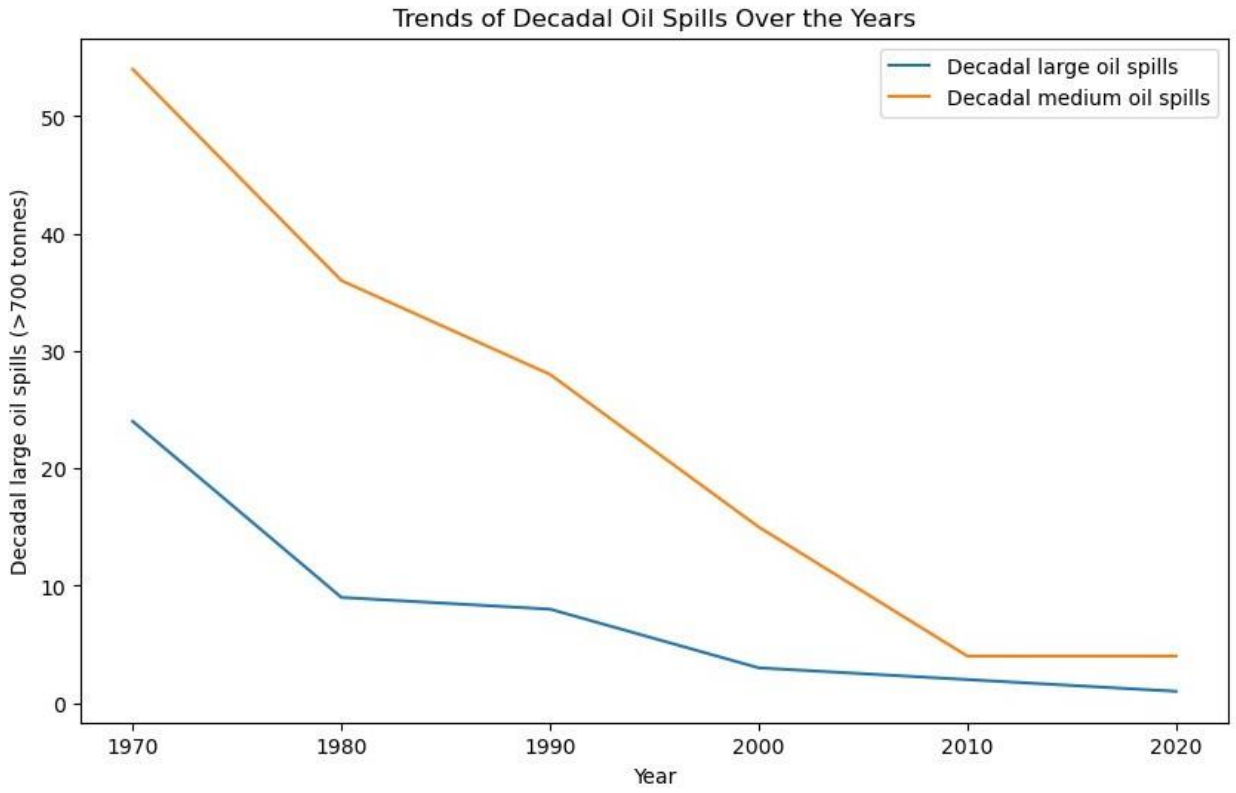
```
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
#Analyze the trends of decadal large and medium oil spills
plt.figure(figsize=(10,6))
sns.lineplot(data=large_oil_spills_decadal,x='Year',y='Decadal large
oil spills (>700 tonnes)',label='Decadal large oil spills')
sns.lineplot(data=large_oil_spills_decadal,x='Year',y='Decadal medium
oil spills (7-700 tonnes)',label='Decadal medium oil spills')
plt.title('Trends of Decadal Oil Spills Over the Years')
plt.show()
```

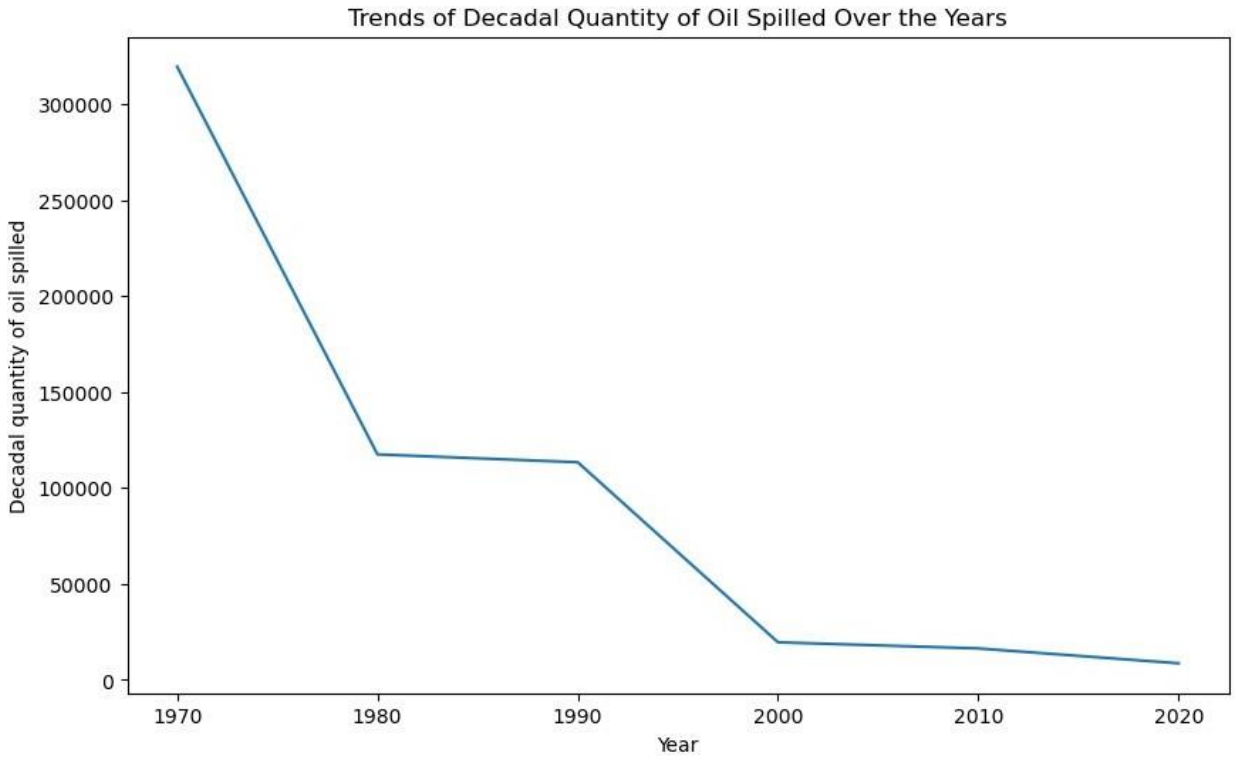
```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
```

```
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```




```
#Analyze the trends of decadal quantity of oil spilled
plt.figure(figsize=(10,6))
sns.lineplot(data=quantity_oil_spills_decadal,x='Year',y='Decadal
quantity of oil spilled')
plt.title('Trends of Decadal Quantity of Oil Spilled Over the Years')
plt.show()

/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed
in a future version. Convert inf values to NaN before operating
```

```
instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



Insights and Recommendations

Based on our trend analysis,we will come up with insights how effectively safety measures and regulations have been over timeand provide some recommendations for how these safety and regulations could be improved.

Insights

After analyzing the data,here's what i've found:

1. It's clear that both large and medium oil spills have been declining over the years.This is a positive sign and suggests that our safety measures and regulations are doing great.

2. Most number of oil spills has occured in 1974.

3.Least number of oil spills has occured in 2019.

4.Most quantity of oil spill has occured in 1979.

5.1970 is the decade where maximum oil spill occured.

6.The decline is not a short term phenomenon but it is progressing over decades which is fantastic.

Recommendations

Based on these insights here's what we've to do next:

1. the safety measures are effective but that doesn't mean we can't make them even better.

2. We need to focus on the causes that are most often linked to oil spills.If we can make safety improvements in these areas ,we could see a big reduction in spills.

3.I recommend we even dig deeper into data and find why our safety measures are working and look for opportunities to make the, even more effective.We are making great progress,but thete's always room for improvement.