```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
pd.set_option('display.max_columns', 50)
pd.set_option('display.max_rows', 50)
```

```
from utility import *
from trends import *
from plot import *
from kpi_computation import *
from forecast import *
```

1. The features in the dataset are continuous. They are as follows:
    - Power consumed by different components
    - Factors influencing power consumption
    - Time series in an interval of 5 minutes for 2 vessels, spanning across a year. That makes it $12*24*365 = 105120$ data points for each vessel.

```
# Read the data
df = pd.read_csv('data/data.csv', header = 0)
df.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Sea Temperature (Celsius)	Boiler 1 Fuel Flow Rate (L/h)	Boiler 2 Fuel Flow Rate (L/h)	Incinerato 1 Fuel Flow Rate (L/h)
0	2023-01-01T00:00:00	2023-01-01T00:05:00	Vessel 1	0.0946	0.1384	5.4654	0.5074	0.0	0.4979	0.4191	27.3000	0.0000	0.0	19.0090
1	2023-01-01T00:05:00	2023-01-01T00:10:00	Vessel 1	0.0540	0.1370	5.4387	0.5158	0.0	0.4982	0.4204	27.3000	47.7695	0.0	216.3180
2	2023-01-01T00:10:00	2023-01-01T00:15:00	Vessel 1	0.0439	0.1785	5.5265	0.5117	0.0	0.5032	0.4199	27.3000	77.2034	0.0	439.4300
3	2023-01-01T00:15:00	2023-01-01T00:20:00	Vessel 1	0.0733	0.1725	5.5257	0.5177	0.0	0.5103	0.4188	27.3076	60.6369	0.0	218.2797
4	2023-01-01T00:20:00	2023-01-01T00:25:00	Vessel 1	0.0780	0.1397	5.4634	0.5169	0.0	0.5100	0.4203	27.3518	55.2184	0.0	0.0000

```
df.describe()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Sea Temperature (Celsius)	Boiler 1 Fu Flow Rate (L/h)
count	210224.000000	210224.000000	210222.000000	210033.000000	210033.000000	210033.000000	210224.000000	210224.000000	210224.000
mean	0.037829	0.118840	4.923284	0.182571	0.159086	0.154729	0.394463	18.304904	36.068139
std	0.164684	0.182357	1.014741	0.236498	0.221811	0.230015	0.298189	6.259071	65.686900
min	0.000000	0.000000	-0.040000	0.000000	0.000000	0.000000	0.000000	2.800000	0.000000
25%	0.007300	0.051600	4.239700	0.000000	0.000000	0.000000	0.155500	13.700000	0.000000
50%	0.025600	0.104300	4.916700	0.000000	0.000000	0.000000	0.321150	18.203900	0.000000
75%	0.055800	0.178500	5.526600	0.386400	0.348900	0.338900	0.763600	21.796525	58.300000
max	40.285400	40.305400	15.264000	0.826100	0.793000	6.305300	1.031500	31.611500	482.057000

```
# Check the data types and column names
df.dtypes
```

```
Start Time                             object
End Time                               object
Vessel Name                            object
Power Galley 1 (MW)                   float64
Power Galley 2 (MW)                   float64
Power Service (MW)                    float64
HVAC Chiller 1 Power (MW)             float64
HVAC Chiller 2 Power (MW)             float64
HVAC Chiller 3 Power (MW)             float64
Scrubber Power (MW)                   float64
Sea Temperature (Celsius)             float64
Boiler 1 Fuel Flow Rate (L/h)         float64
Boiler 2 Fuel Flow Rate (L/h)         float64
Incinerator 1 Fuel Flow Rate (L/h)    float64
Diesel Generator 1 Power (MW)         float64
Diesel Generator 2 Power (MW)         float64
Diesel Generator 3 Power (MW)         float64
Diesel Generator 4 Power (MW)         float64
Latitude (Degrees)                    float64
Longitude (Degrees)                   float64
Relative Wind Angle (Degrees)         float64
True Wind Angle (Degrees)             float64
Depth (m)                             float64
Relative Wind Direction (Degrees)     float64
True Wind Direction (Degrees)         float64
Draft (m)                             float64
Speed Over Ground (knots)             float64
True Wind Speed (knots)               float64
Relative Wind Speed (knots)           float64
Speed Through Water (knots)           float64
Local Time (h)                        float64
Trim (m)                              float64
Propulsion Power (MW)                 float64
Port Side Propulsion Power (MW)       float64
Starboard Side Propulsion Power (MW)  float64
Bow Thruster 1 Power (MW)             float64
Bow Thruster 2 Power (MW)             float64
Bow Thruster 3 Power (MW)             float64
Stern Thruster 1 Power (MW)           float64
Stern Thruster 2 Power (MW)           float64
Main Engine 1 Fuel Flow Rate (kg/h)   float64
Main Engine 2 Fuel Flow Rate (kg/h)   float64
Main Engine 3 Fuel Flow Rate (kg/h)   float64
Main Engine 4 Fuel Flow Rate (kg/h)   float64
dtype: object
```

The features in the entire dataset are (without the Start time, End Time and the Vessel Name):

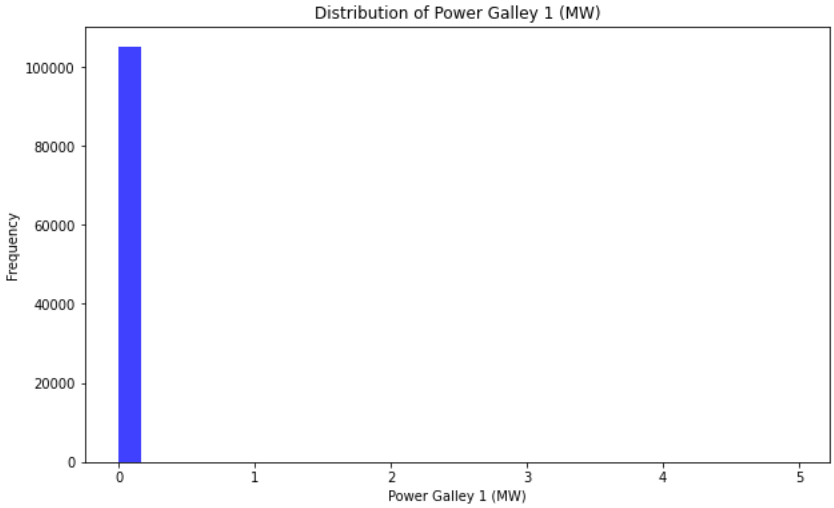
Column Name
Power Galley 1 (MW)

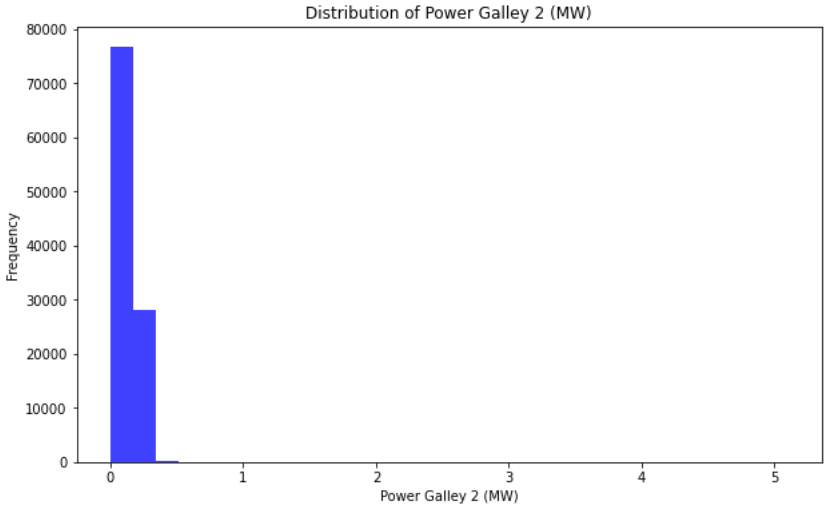
Column Name
Power Galley 2 (MW)
Power Service (MW)
HVAC Chiller 1 Power (MW)
HVAC Chiller 2 Power (MW)
HVAC Chiller 3 Power (MW)
Scrubber Power (MW)
Sea Temperature (Celsius)
Boiler 1 Fuel Flow Rate (L/h)
Boiler 2 Fuel Flow Rate (L/h)
Incinerator 1 Fuel Flow Rate (L/h)
Diesel Generator 1 Power (MW)
Diesel Generator 2 Power (MW)
Diesel Generator 3 Power (MW)
Diesel Generator 4 Power (MW)
Latitude (Degrees)
Longitude (Degrees)
Relative Wind Angle (Degrees)
True Wind Angle (Degrees)
Depth (m)
Relative Wind Direction (Degrees)
True Wind Direction (Degrees)
Draft (m)
Speed Over Ground (knots)
True Wind Speed (knots)
Relative Wind Speed (knots)
Speed Through Water (knots)
Local Time (h)
Trim (m)
Propulsion Power (MW)
Port Side Propulsion Power (MW)
Starboard Side Propulsion Power (MW)
Bow Thruster 1 Power (MW)
Bow Thruster 2 Power (MW)
Bow Thruster 3 Power (MW)
Stern Thruster 1 Power (MW)
Stern Thruster 2 Power (MW)
Main Engine 1 Fuel Flow Rate (kg/h)
Main Engine 2 Fuel Flow Rate (kg/h)
Main Engine 3 Fuel Flow Rate (kg/h)
Main Engine 4 Fuel Flow Rate (kg/h)

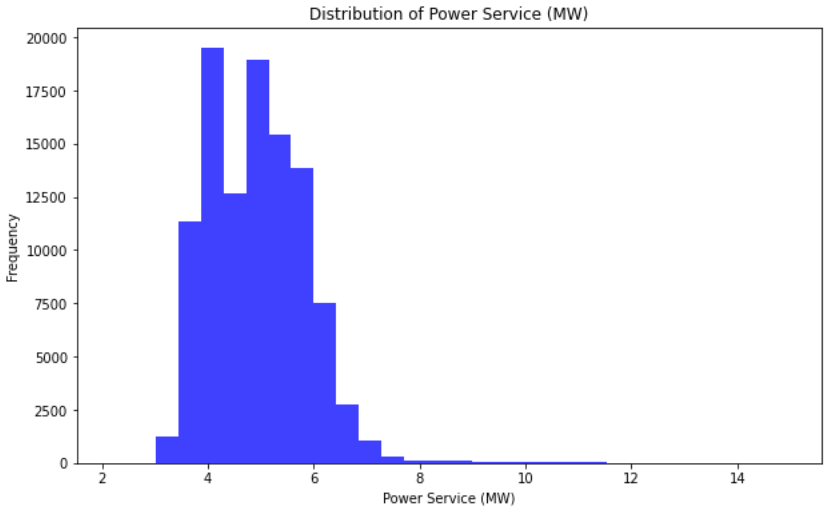
Vessel - Level Analysis

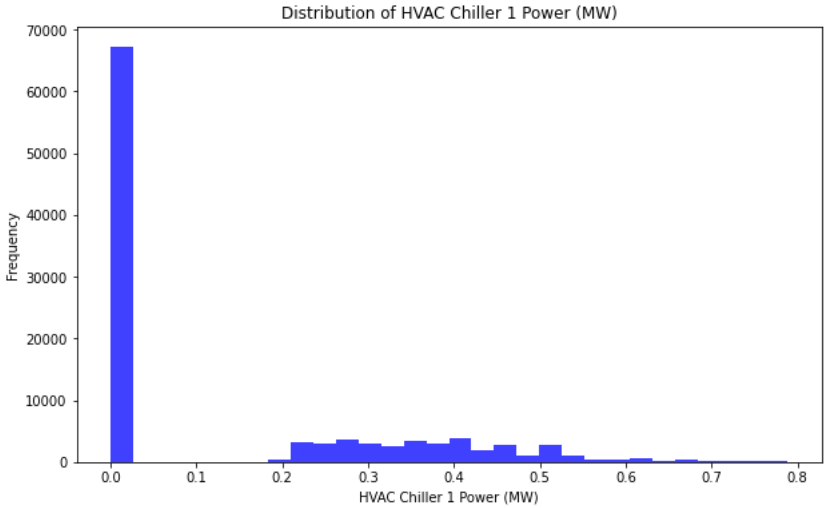
```
dfv = pick_vessel(df, 'Vessel 1')
# dfv = pick_vessel(df, 'Vessel 2')
# dfv = df
```

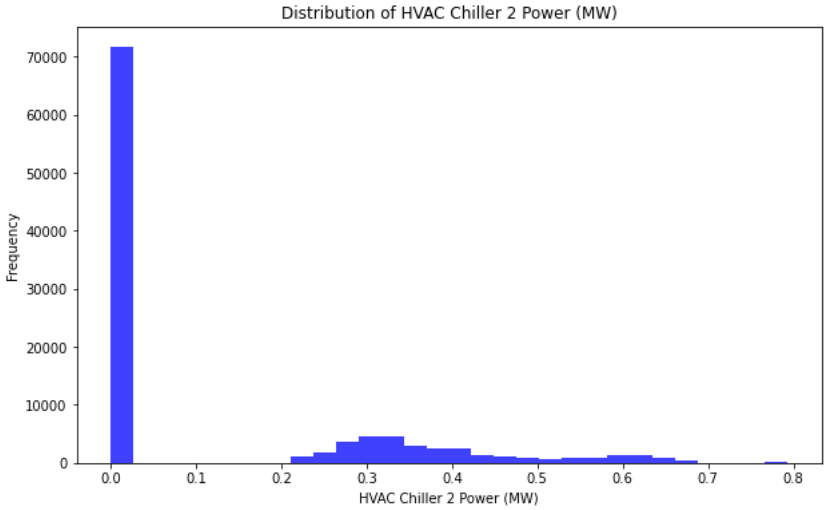
```
# Generates the distribution of the features in the dataset as histogram plots
feature_distribution(dfv)
```

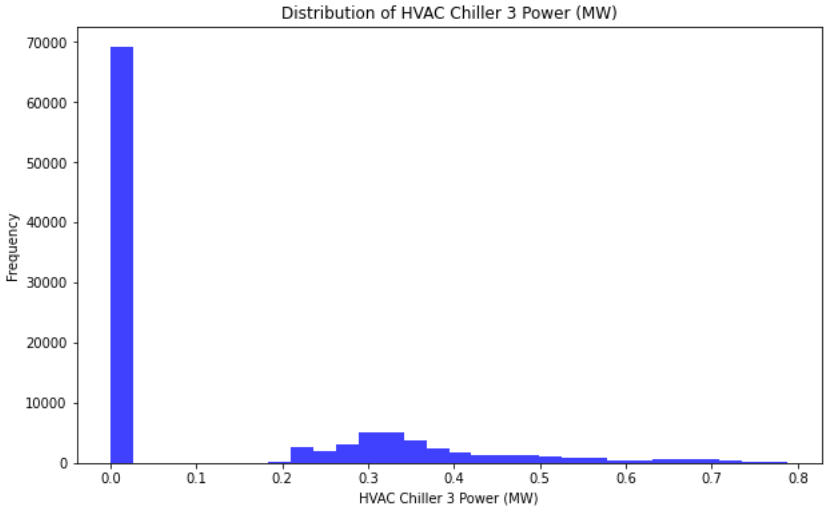


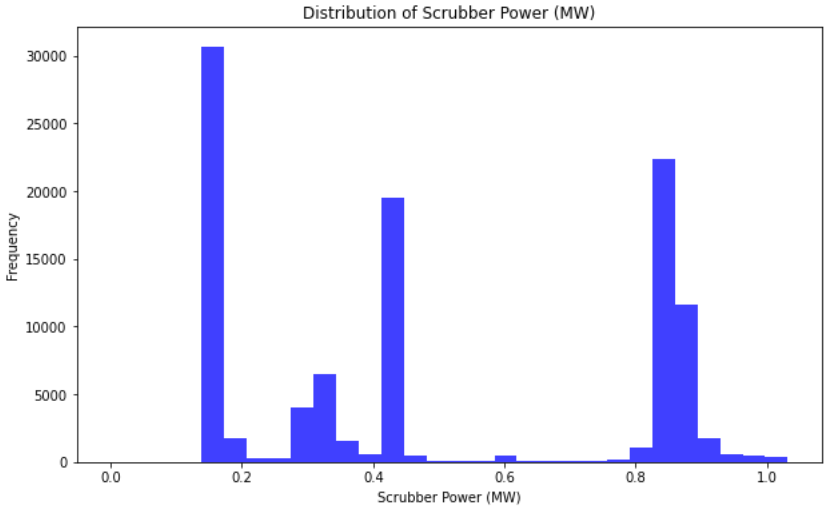


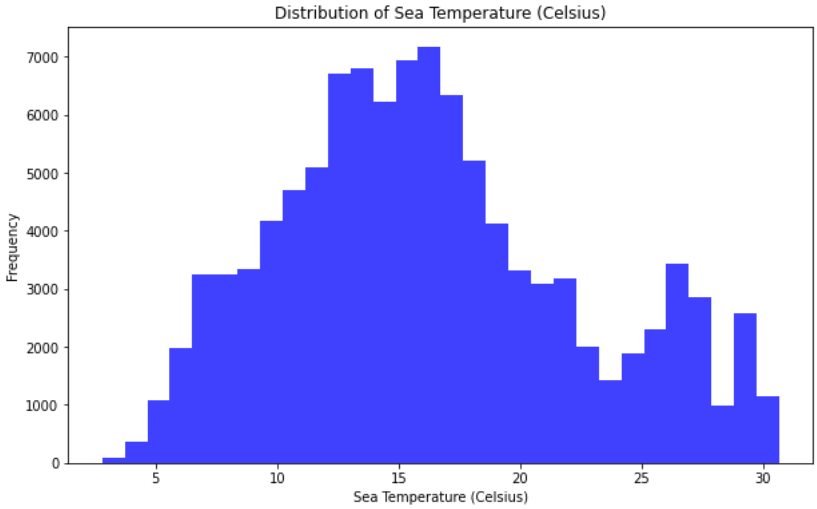


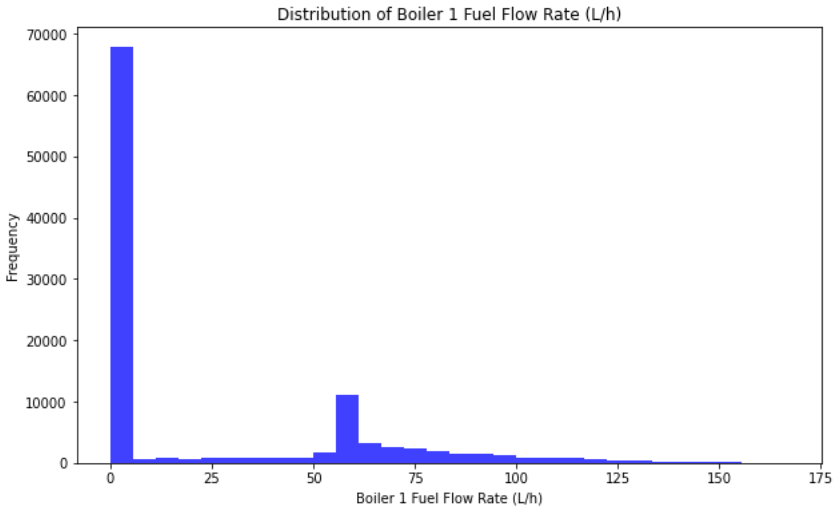

Distribution of HVAC Chiller 1 Power (MW)
Frequency
HVAC Chiller 1 Power (MW)

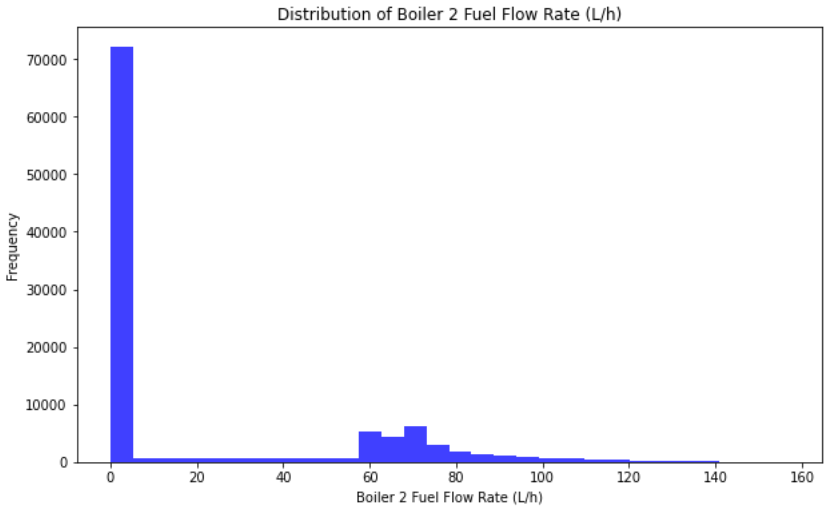

Distribution of HVAC Chiller 2 Power (MW)
Frequency
HVAC Chiller 2 Power (MW)

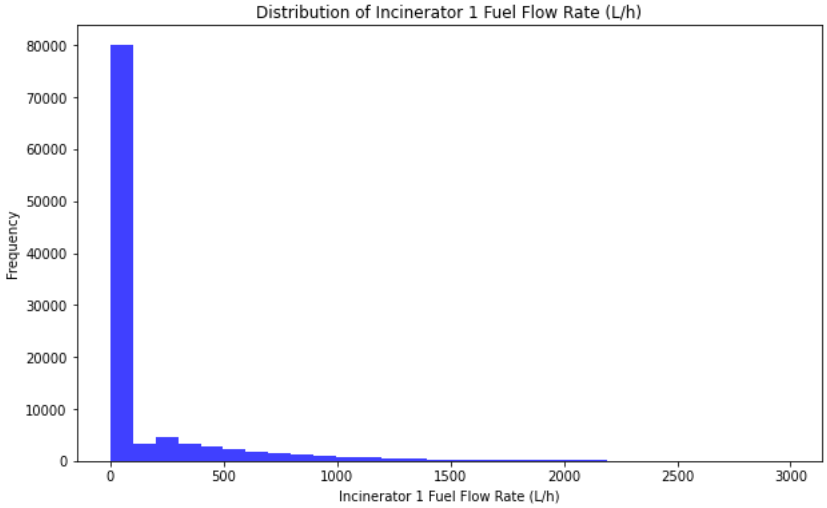

Distribution of HVAC Chiller 3 Power (MW)
Frequency
HVAC Chiller 3 Power (MW)

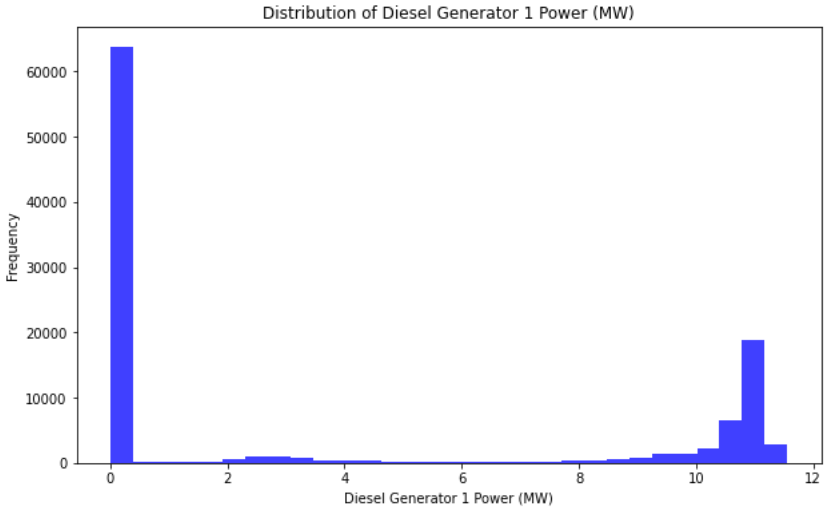

Distribution of Scrubber Power (MW)
Frequency
Scrubber Power (MW)
0
5000
10000
15000
20000
25000
30000
0.0
0.2
0.4
0.6
0.8
1.0

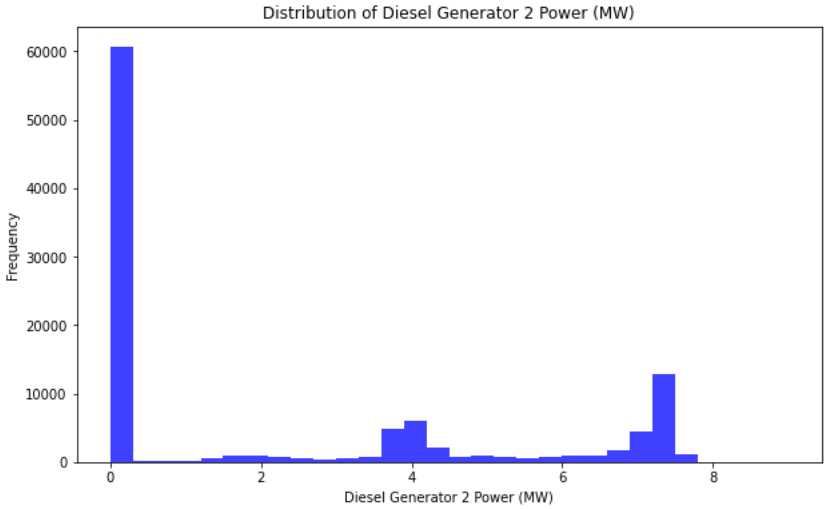

Distribution of Sea Temperature (Celsius)
Frequency
Sea Temperature (Celsius)
0
1000
2000
3000
4000
5000
6000
7000
5
10
15
20
25
30

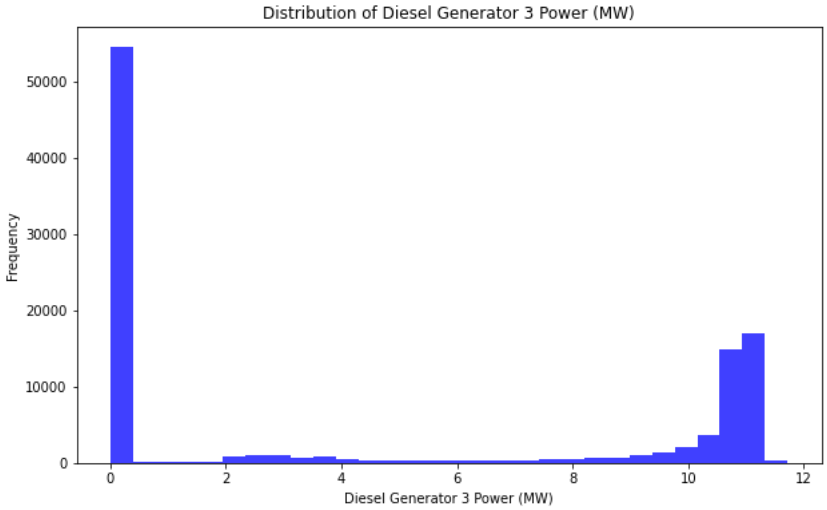

Distribution of Boiler 1 Fuel Flow Rate (L/h)
Frequency
Boiler 1 Fuel Flow Rate (L/h)
0
10000
20000
30000
40000
50000
60000
70000
0
25
50
75
100
125
150
175

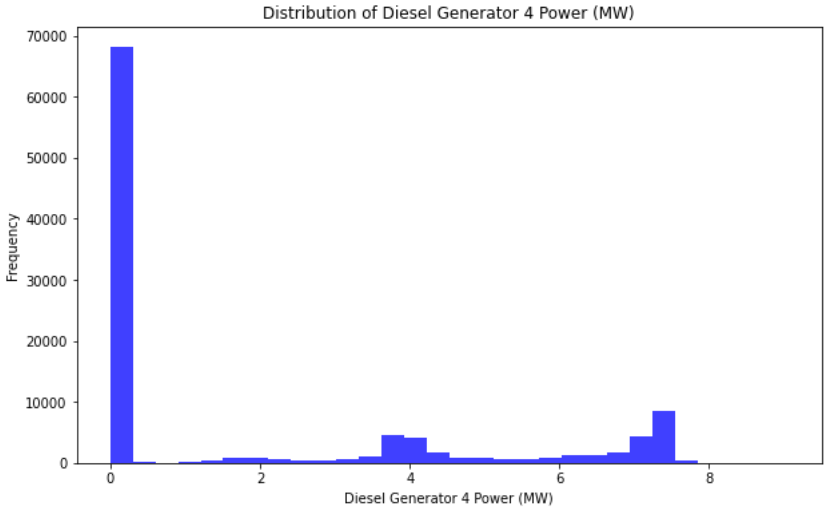

Distribution of Boiler 2 Fuel Flow Rate (L/h)
Frequency
Boiler 2 Fuel Flow Rate (L/h)
0
10000
20000
30000
40000
50000
60000
70000
0
20
40
60
80
100
120
140
160

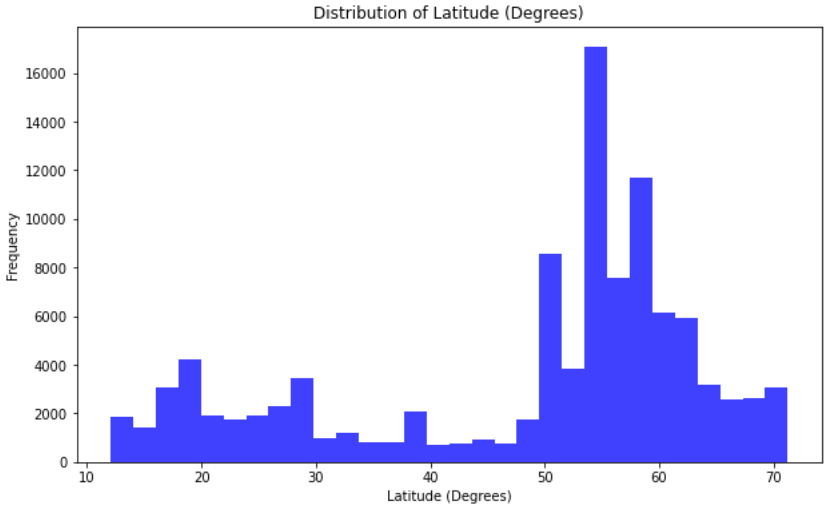

Distribution of Incinerator 1 Fuel Flow Rate (L/h)
Frequency
Incinerator 1 Fuel Flow Rate (L/h)
0
10000
20000
30000
40000
50000
60000
70000
80000
0
500
1000
1500
2000
2500
3000

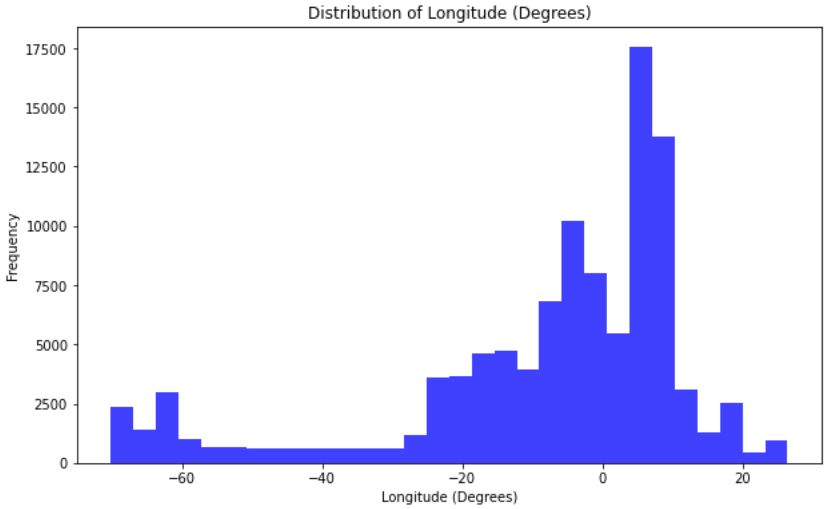

Distribution of Diesel Generator 1 Power (MW)
Frequency
Diesel Generator 1 Power (MW)
0
10000
20000
30000
40000
50000
60000
0
2
4
6
8
10
12

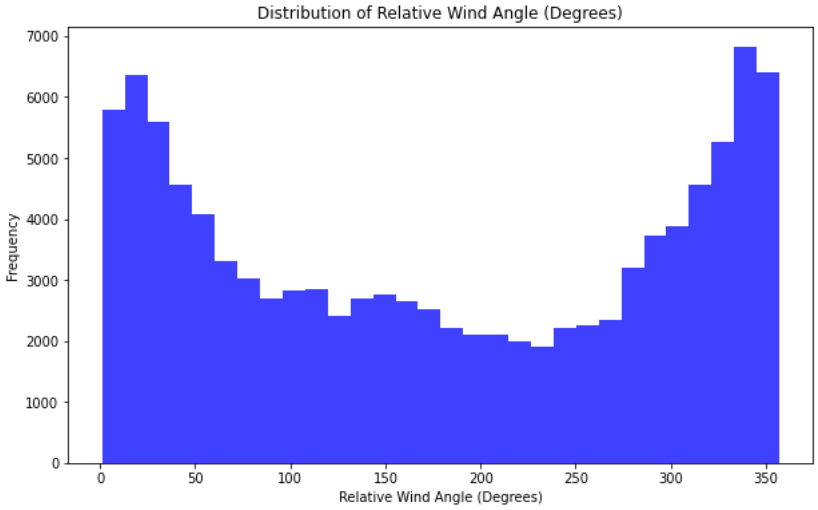

Distribution of Diesel Generator 2 Power (MW)
Frequency
Diesel Generator 2 Power (MW)

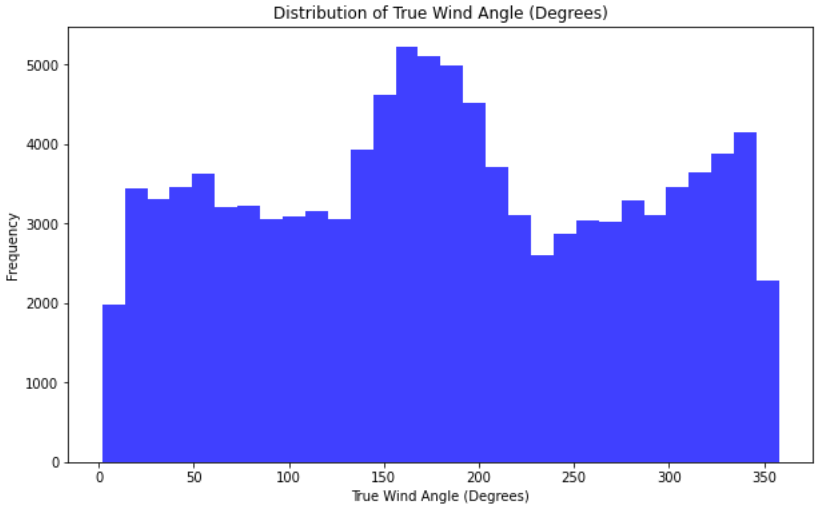

Distribution of Diesel Generator 3 Power (MW)
Frequency
Diesel Generator 3 Power (MW)

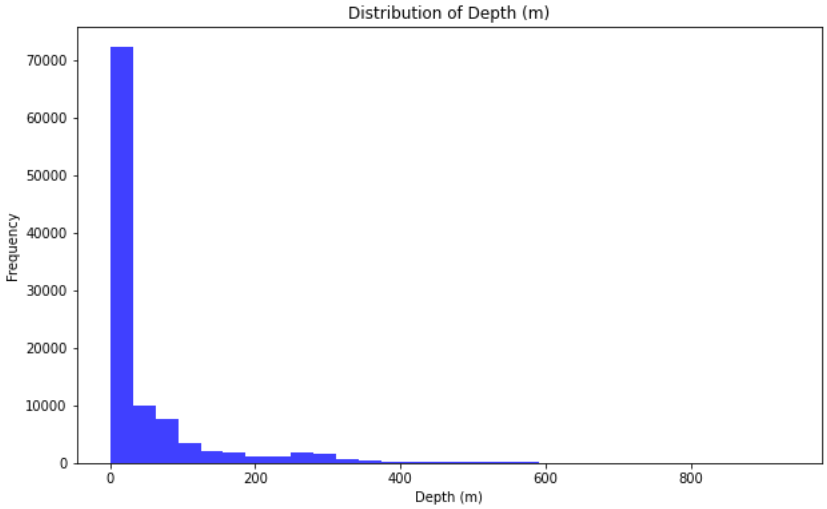

Distribution of Diesel Generator 4 Power (MW)
Frequency
Diesel Generator 4 Power (MW)

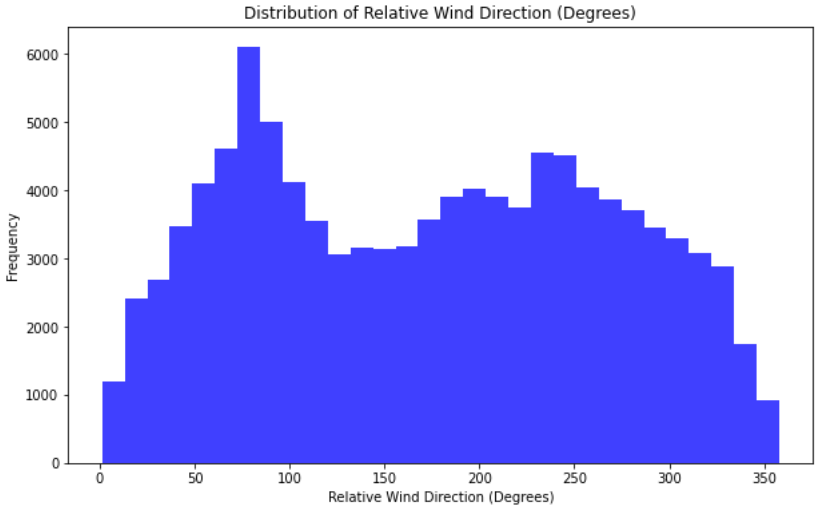

Distribution of Latitude (Degrees)
Frequency
Latitude (Degrees)

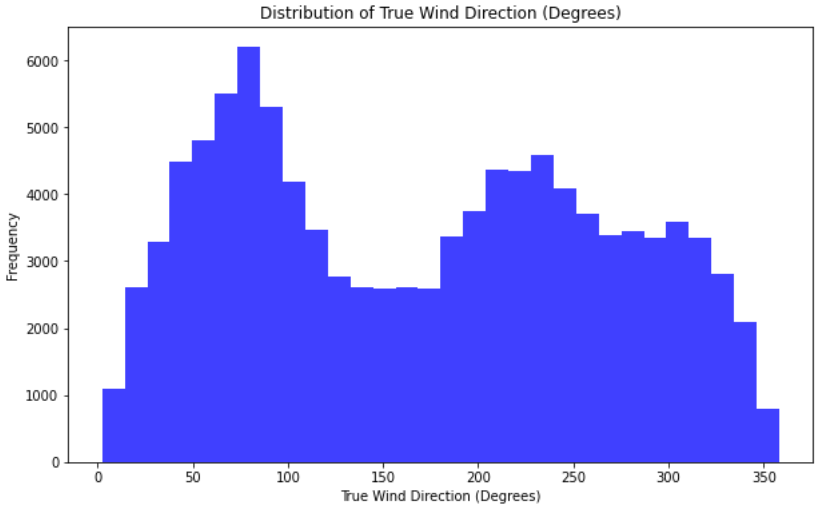

Distribution of Longitude (Degrees)
Frequency
Longitude (Degrees)

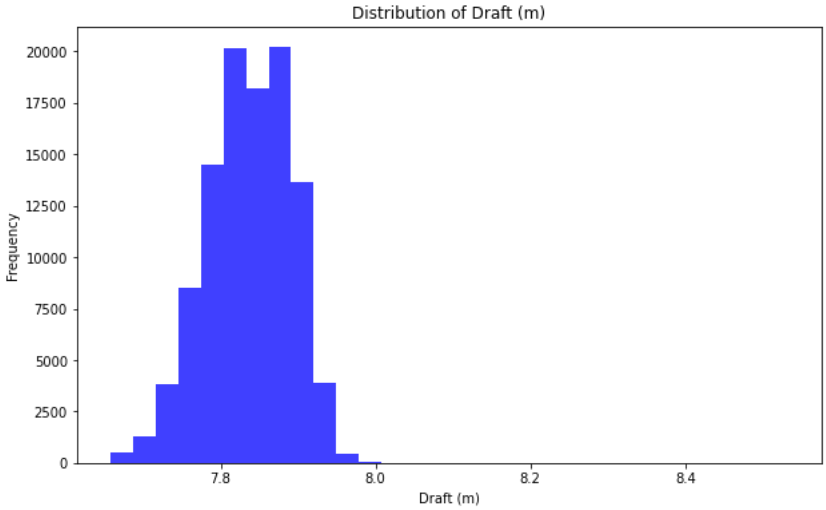

Distribution of Relative Wind Angle (Degrees)
Frequency
Relative Wind Angle (Degrees)

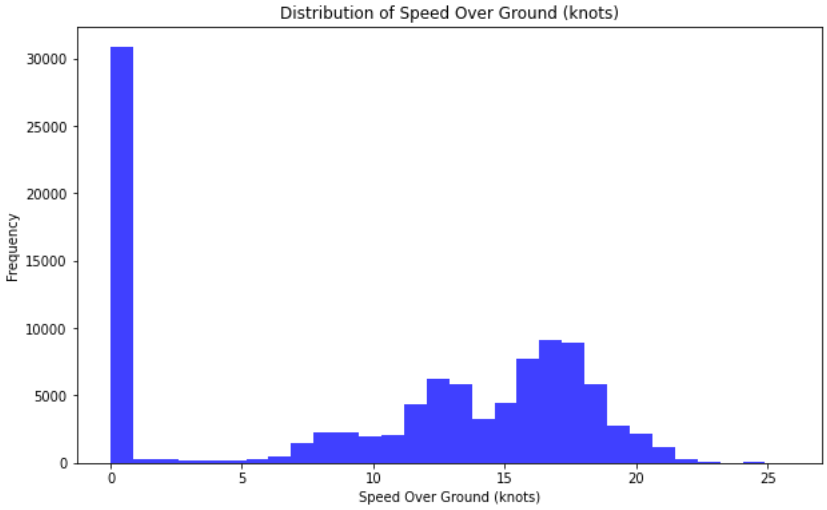

Distribution of True Wind Angle (Degrees)
Frequency
True Wind Angle (Degrees)
0
1000
2000
3000
4000
5000
0
50
100
150
200
250
300
350

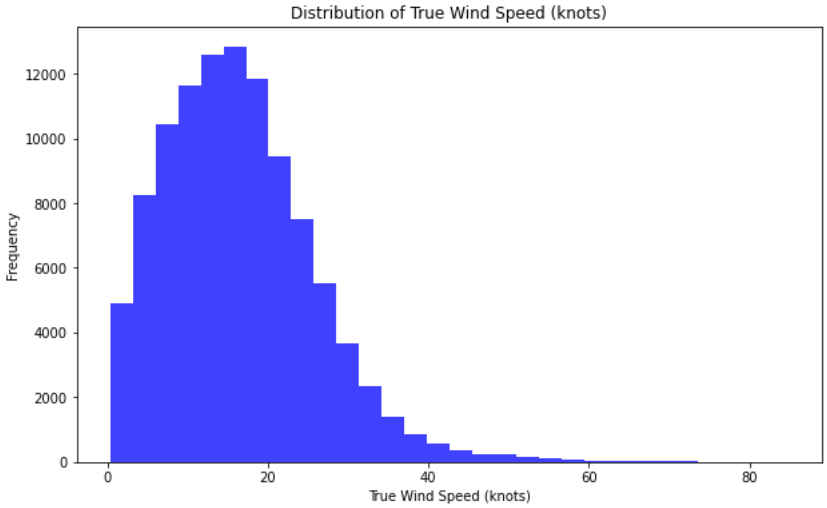

Distribution of Depth (m)
Frequency
Depth (m)
0
10000
20000
30000
40000
50000
60000
70000
0
200
400
600
800

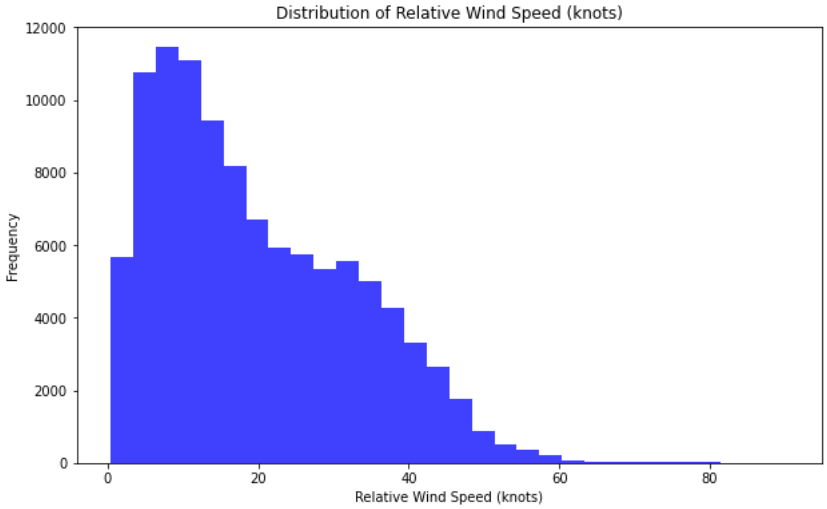

Distribution of Relative Wind Direction (Degrees)
Frequency
Relative Wind Direction (Degrees)
0
1000
2000
3000
4000
5000
6000
0
50
100
150
200
250
300
350

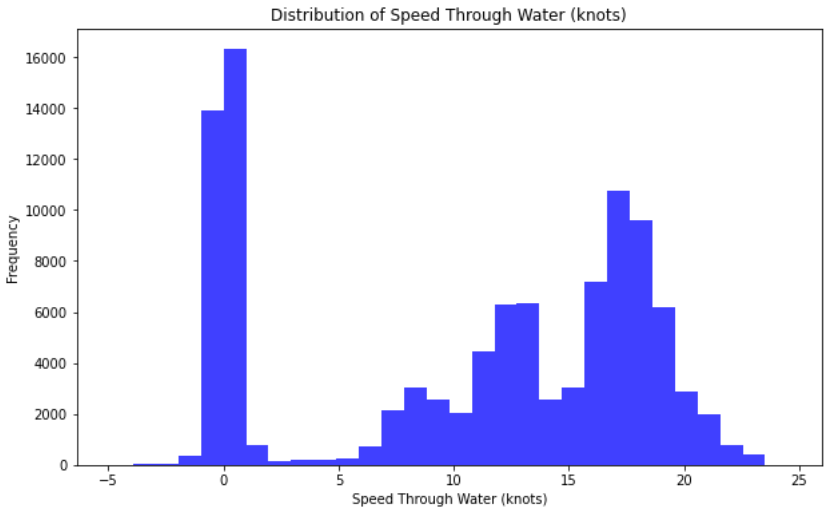

Distribution of True Wind Direction (Degrees)
Frequency
True Wind Direction (Degrees)
0
1000
2000
3000
4000
5000
6000
0
50
100
150
200
250
300
350

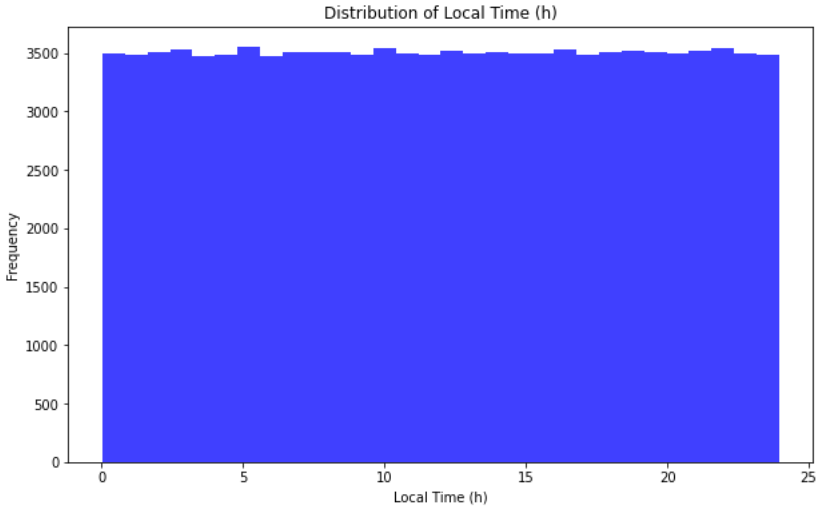

Distribution of Draft (m)
Frequency
Draft (m)
0
2500
5000
7500
10000
12500
15000
17500
20000
7.8
8.0
8.2
8.4

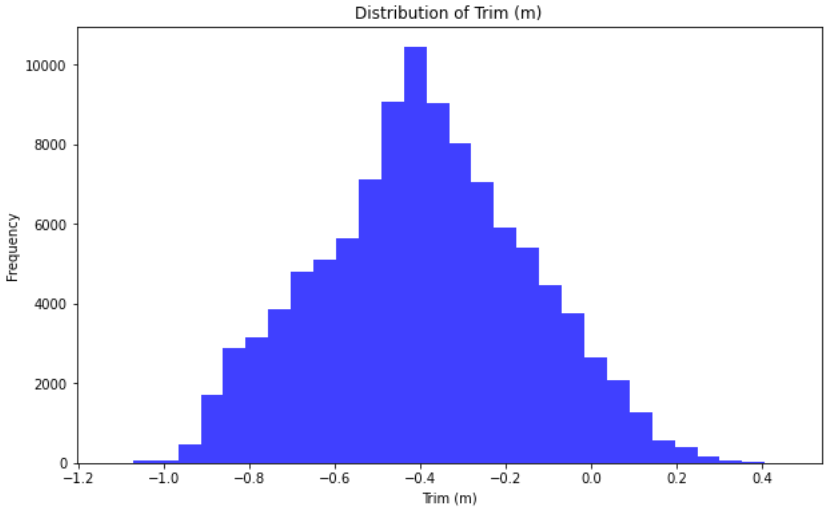

Distribution of Speed Over Ground (knots)
Frequency
Speed Over Ground (knots)
0
5000
10000
15000
20000
25000
30000
0
5
10
15
20
25

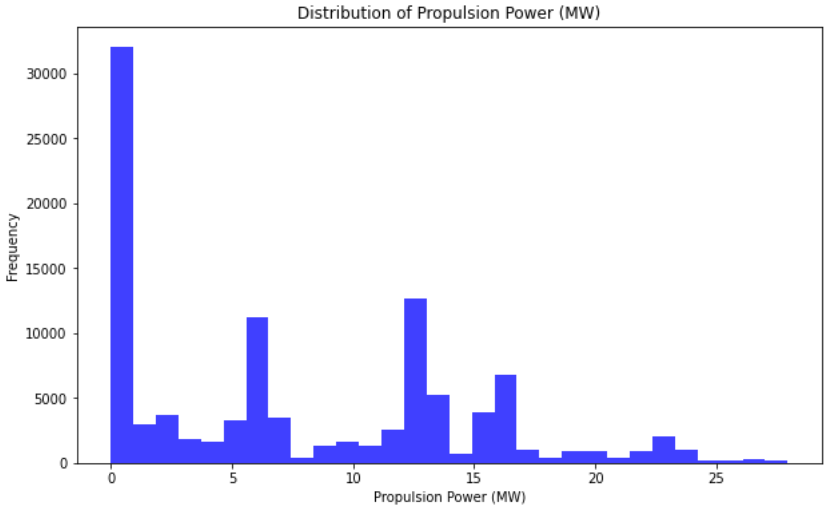

Distribution of True Wind Speed (knots)
Frequency
True Wind Speed (knots)
0
2000
4000
6000
8000
10000
12000
0
20
40
60
80

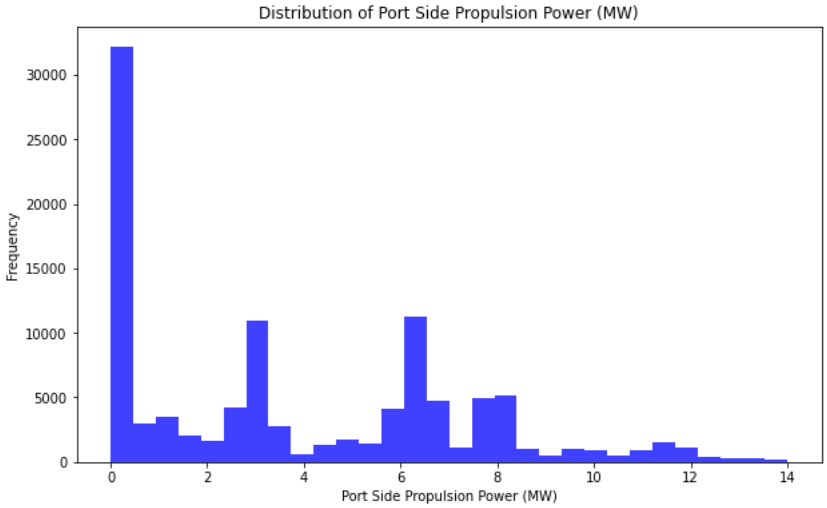

Distribution of Relative Wind Speed (knots)
Frequency
Relative Wind Speed (knots)
0
2000
4000
6000
8000
10000
12000
0
20
40
60
80

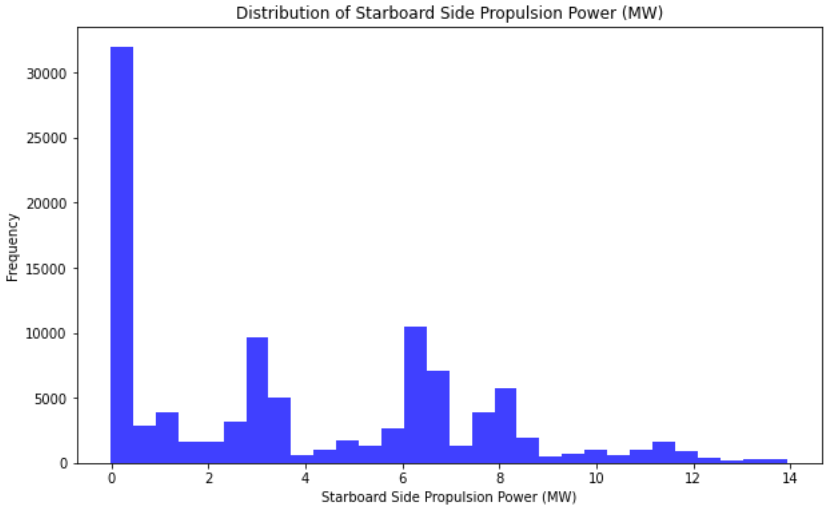

Distribution of Speed Through Water (knots)
Frequency
Speed Through Water (knots)
0
2000
4000
6000
8000
10000
12000
14000
16000
−5
0
5
10
15
20
25

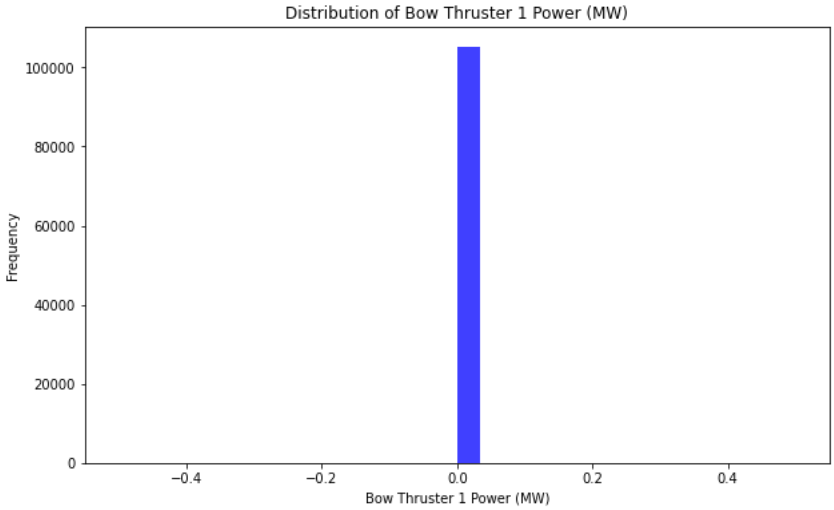

Distribution of Local Time (h)
Frequency
Local Time (h)
0
500
1000
1500
2000
2500
3000
3500
0
5
10
15
20
25

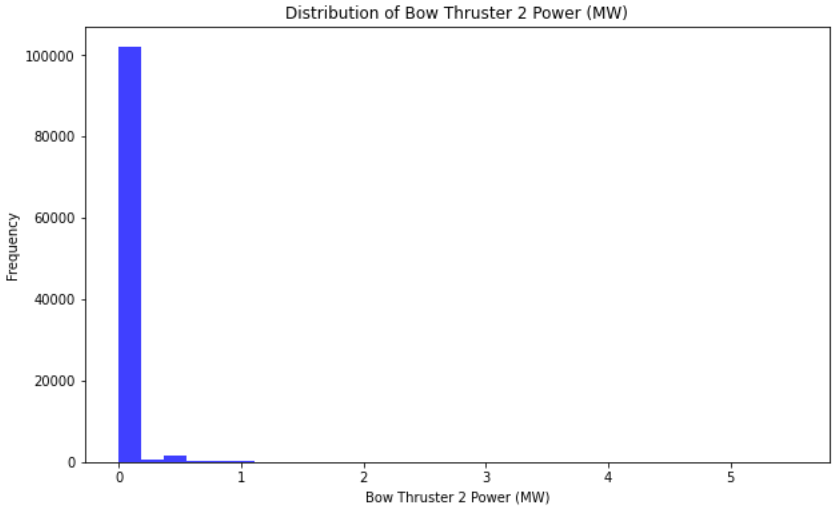

Distribution of Trim (m)
Frequency
Trim (m)
0
2000
4000
6000
8000
10000
−1.2
−1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4

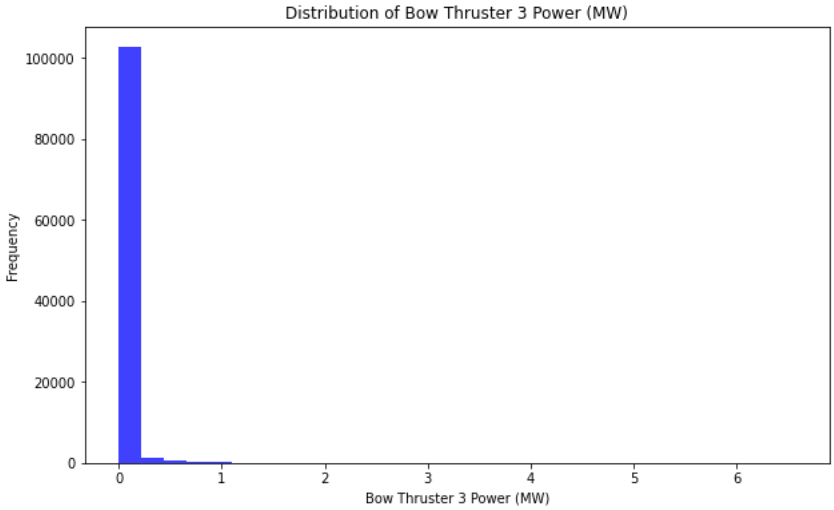

Distribution of Propulsion Power (MW)
Frequency
Propulsion Power (MW)
0
5000
10000
15000
20000
25000
30000
0
5
10
15
20
25

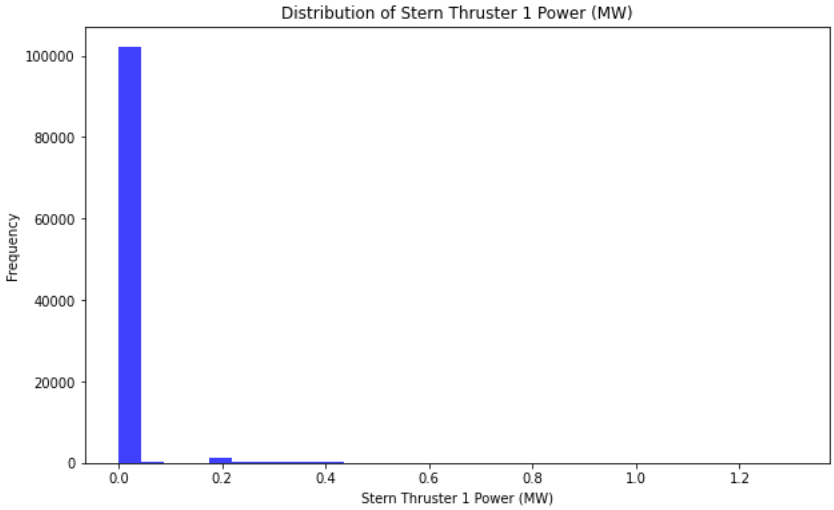

Distribution of Port Side Propulsion Power (MW)
Frequency
Port Side Propulsion Power (MW)
0
5000
10000
15000
20000
25000
30000
0
2
4
6
8
10
12
14

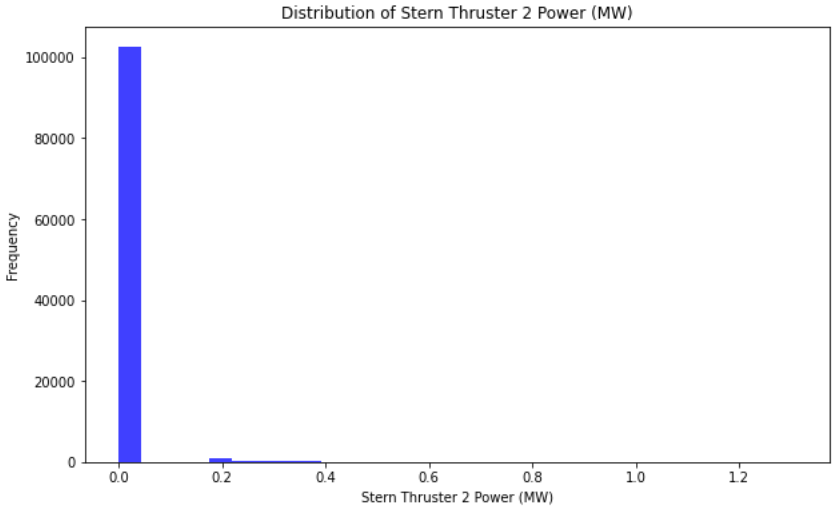

Distribution of Starboard Side Propulsion Power (MW)
Frequency
Starboard Side Propulsion Power (MW)
0
5000
10000
15000
20000
25000
30000
0
2
4
6
8
10
12
14

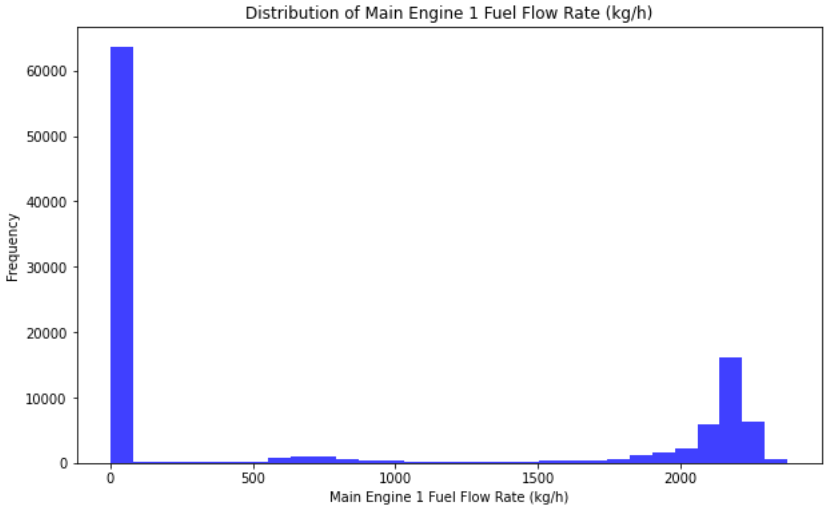

Distribution of Bow Thruster 1 Power (MW)
Frequency
Bow Thruster 1 Power (MW)
0
20000
40000
60000
80000
100000
−0.4
−0.2
0.0
0.2
0.4

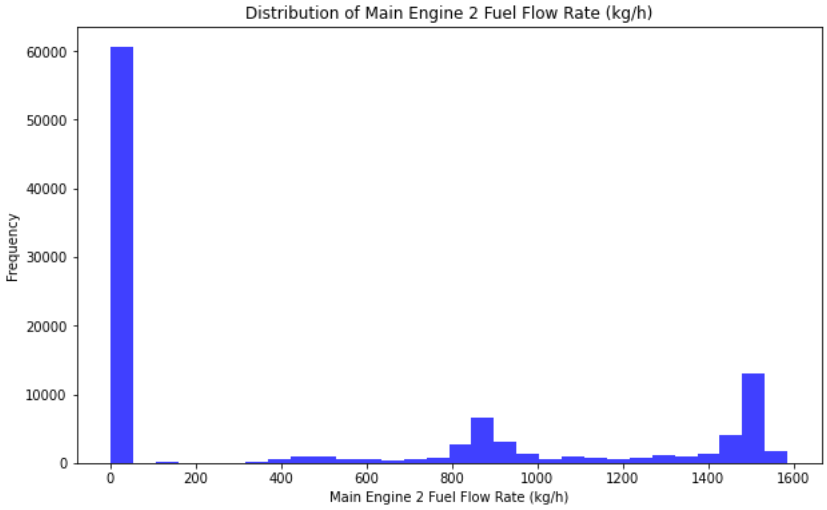

Distribution of Bow Thruster 2 Power (MW)
Frequency
Bow Thruster 2 Power (MW)
0
20000
40000
60000
80000
100000
0
1
2
3
4
5

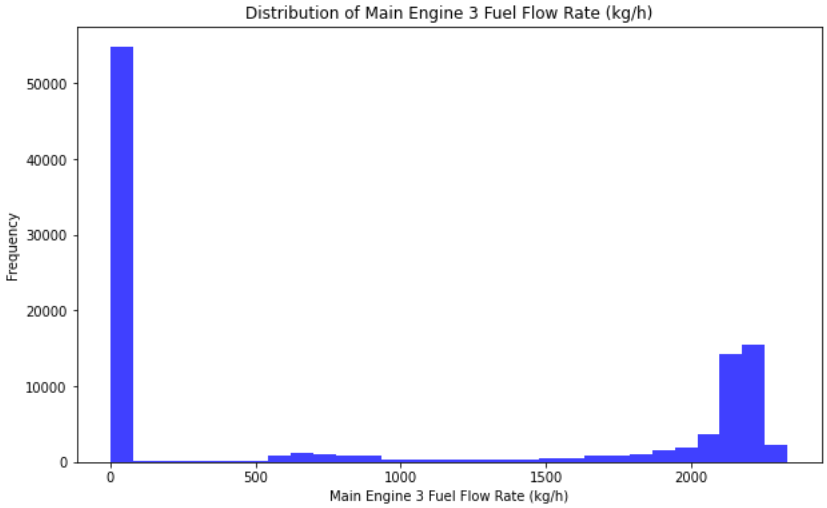

Distribution of Bow Thruster 3 Power (MW)
Frequency
Bow Thruster 3 Power (MW)
0
20000
40000
60000
80000
100000
0
1
2
3
4
5
6

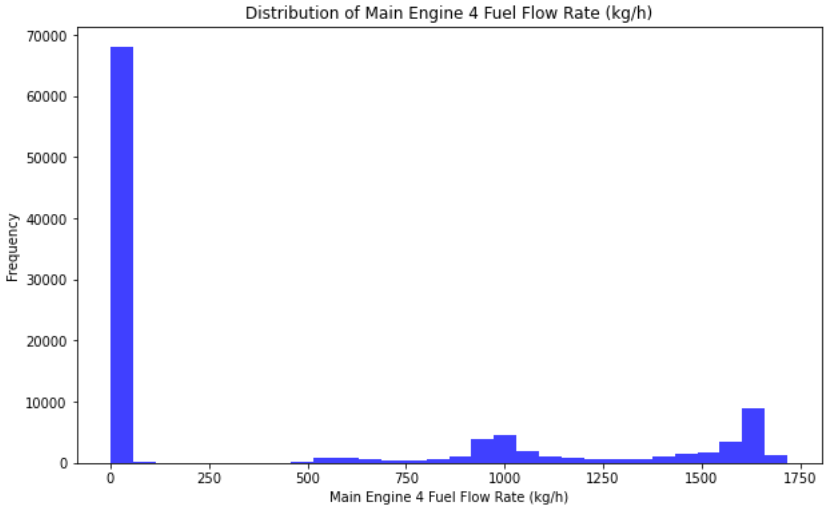

Distribution of Stern Thruster 1 Power (MW)
Frequency
Stern Thruster 1 Power (MW)
0
20000
40000
60000
80000
100000
0.0
0.2
0.4
0.6
0.8
1.0
1.2


Distribution of Stern Thruster 2 Power (MW)
Frequency
Stern Thruster 2 Power (MW)
0
20000
40000
60000
80000
100000
0.0
0.2
0.4
0.6
0.8
1.0
1.2


Distribution of Main Engine 1 Fuel Flow Rate (kg/h)
Frequency
Main Engine 1 Fuel Flow Rate (kg/h)
0
10000
20000
30000
40000
50000
60000
0
500
1000
1500
2000


Distribution of Main Engine 2 Fuel Flow Rate (kg/h)
Frequency
Main Engine 2 Fuel Flow Rate (kg/h)
0
10000
20000
30000
40000
50000
60000
0
200
400
600
800
1000
1200
1400
1600





Comments (Vessel 1):

1. Power Galleys: Mostly constant range of power consumption in operation. Less variation and wouldn't contribute much in forecast models
2. HVAC Chillers: A lot of variation, but extremely skewed in the sense that most of of the operation range is fixed
3. Scrubber Power: Shows 4 peaks, denoting 4 levels of high intensity cleaning activities
4. Power Service: Service Power is more uniformly distributed but doesn't have much variation in the data
5. Sea Temperature: Primarily influenced by the season of the year, and equitorial position of the ship, is uniformly distributed except for a peak on the right side indicating sligh skewness probably due to stronger summer season in one of the locations
6. Boiler Fuel Flow: Consists of a tall box and a tiny peak at the higher end of fuel flow rate. Most of the usual operations happen in between 0-10L/h fuel flow. Rarely, on days of extreme weather or rough operations
7. Incinerator: Majority of the waste removal operation happens with a lower fuel requirement, except for some extreme cases. These could be during docking for extra cleaning/occassional discharge cleaning.
8. Diesel Generators: Substantiate a 2-3 peaks highlighting a 2-3 modes of operation, including traveling against currents/winds requiring higher throughput
9. Latitude and Longitude: Positions indicate travel in the northern hemisphere, around Atlantic, with extra time spent close to the prime meridian
10. Relative Wind angle: Indicates that the cruise ship tries to move and orient itself in-line to the wind 0deg or 360deg for conserving energy by taking the momentum from wind and also by reducing drag
11. Depth: The cruise tries to maintain consistent depth throughout between 0-100 metres
12. Draft: maintained constantly between 7.5 and 8 metres, shows buoyancy control and build quality of the ship
13. Relative Wind Speed: The wind speed is aligned in favor to the ship movement by aligning the direction of sail
14. Speed through water/over land: Indicates 2 modes, one low rangee where the ship is either docking, anchoring or sailing in shallow waters. The second peak which is not so frequent indicates high speed motion. Among these, Speed through water has higher peak on greater end of speed spectrum because the ship has to move against the water many a times.
15. Trim: A uniform and normally distributed trim is maintained indicating well-planned design execution throughout the operation
16. Propulsion power has 4 peaks, indicating 4 ranges of operation. It is well spread out with variation, higher power being demanded probably when the ship is leaving from docks and is on shallow waters.
17. Thrusters: Limited operating ranges with 0-0.1 MW consumed in maneuvering primarily
18. Main Engine Fuel Flow: Is optimized to operate mostly at lower end without many fluctuations. This is indicated by a left skewed histogram, with only rare activities on the higher end

```
missing_values = dfv.isna().sum()
missing_values.plot(kind='bar',figsize=(12,5), title='Missing Values')  # Plot the missing values
```

Impute the missing Values:

1. For features that have <1% missing values, impute by interpolating them using closest non-missing values as the features are all usually smooth within the 5mins time intervals in which they are recorded
2. For features that have >20% missing values, impute by using median of the column/feautre

```
# Imputing the missing values
missing_values
```

```
col_to_interpolate = dfv.columns.difference(['Depth (m)', 'Start Time', 'End Time', 'Vessel Name']) # Columns to interpolate
```

```
impute_time_series(dfv, col_to_interpolate) # Imputation via interpolation for the columns with < 1% missing values as they are likely to be continuous in time
dfv.isna().sum()
```

```
c:\Users\karth\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\core\indexing.py:1773: 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
  self._setitem_single_column(ilocs[0], value, pi)




Start Time                                0
End Time                                  0
Vessel Name                               0
Power Galley 1 (MW)                       0
Power Galley 2 (MW)                       0
Power Service (MW)                        0
HVAC Chiller 1 Power (MW)                 0
HVAC Chiller 2 Power (MW)                 0
HVAC Chiller 3 Power (MW)                 0
Scrubber Power (MW)                       0
Sea Temperature (Celsius)                 0
Boiler 1 Fuel Flow Rate (L/h)             0
Boiler 2 Fuel Flow Rate (L/h)             0
Incinerator 1 Fuel Flow Rate (L/h)        0
Diesel Generator 1 Power (MW)             0
Diesel Generator 2 Power (MW)             0
Diesel Generator 3 Power (MW)             0
Diesel Generator 4 Power (MW)             0
Latitude (Degrees)                        0
Longitude (Degrees)                       0
Relative Wind Angle (Degrees)             0
True Wind Angle (Degrees)                 0
Depth (m)                             27756
Relative Wind Direction (Degrees)         0
True Wind Direction (Degrees)             0
Draft (m)                                 0
Speed Over Ground (knots)                 0
True Wind Speed (knots)                   0
Relative Wind Speed (knots)               0
Speed Through Water (knots)               0
Local Time (h)                            0
Trim (m)                                  0
Propulsion Power (MW)                     0
Port Side Propulsion Power (MW)           0
Starboard Side Propulsion Power (MW)      0
Bow Thruster 1 Power (MW)                 0
Bow Thruster 2 Power (MW)                 0
Bow Thruster 3 Power (MW)                 0
Stern Thruster 1 Power (MW)               0
Stern Thruster 2 Power (MW)               0
Main Engine 1 Fuel Flow Rate (kg/h)       0
Main Engine 2 Fuel Flow Rate (kg/h)       0
Main Engine 3 Fuel Flow Rate (kg/h)       0
```

```
Main Engine 4 Fuel Flow Rate (kg/h)         0
dtype: int64
```

```
median_depth = dfv['Depth (m)'].median() # impution via median value as more than 20% of the values are missing
dfv['Depth (m)'].fillna(median_depth, inplace=True)
dfv.isna().sum()
```

```
c:\Users\karth\AppData\Local\Programs\Python\Python310\lib\site-packages\pandas\core\generic.py:6392: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-
versus-a-copy
  return self._update_inplace(result)
```

```
Start Time                              0
End Time                                0
Vessel Name                             0
Power Galley 1 (MW)                     0
Power Galley 2 (MW)                     0
Power Service (MW)                      0
HVAC Chiller 1 Power (MW)               0
HVAC Chiller 2 Power (MW)               0
HVAC Chiller 3 Power (MW)               0
Scrubber Power (MW)                     0
Sea Temperature (Celsius)               0
Boiler 1 Fuel Flow Rate (L/h)           0
Boiler 2 Fuel Flow Rate (L/h)           0
Incinerator 1 Fuel Flow Rate (L/h)      0
Diesel Generator 1 Power (MW)           0
Diesel Generator 2 Power (MW)           0
Diesel Generator 3 Power (MW)           0
Diesel Generator 4 Power (MW)           0
Latitude (Degrees)                      0
Longitude (Degrees)                     0
Relative Wind Angle (Degrees)           0
True Wind Angle (Degrees)               0
Depth (m)                               0
Relative Wind Direction (Degrees)       0
True Wind Direction (Degrees)           0
Draft (m)                               0
Speed Over Ground (knots)               0
True Wind Speed (knots)                 0
Relative Wind Speed (knots)             0
Speed Through Water (knots)             0
Local Time (h)                          0
Trim (m)                                0
Propulsion Power (MW)                   0
Port Side Propulsion Power (MW)         0
Starboard Side Propulsion Power (MW)    0
Bow Thruster 1 Power (MW)               0
Bow Thruster 2 Power (MW)               0
Bow Thruster 3 Power (MW)               0
Stern Thruster 1 Power (MW)             0
Stern Thruster 2 Power (MW)             0
Main Engine 1 Fuel Flow Rate (kg/h)     0
Main Engine 2 Fuel Flow Rate (kg/h)     0
Main Engine 3 Fuel Flow Rate (kg/h)     0
Main Engine 4 Fuel Flow Rate (kg/h)     0
dtype: int64
```

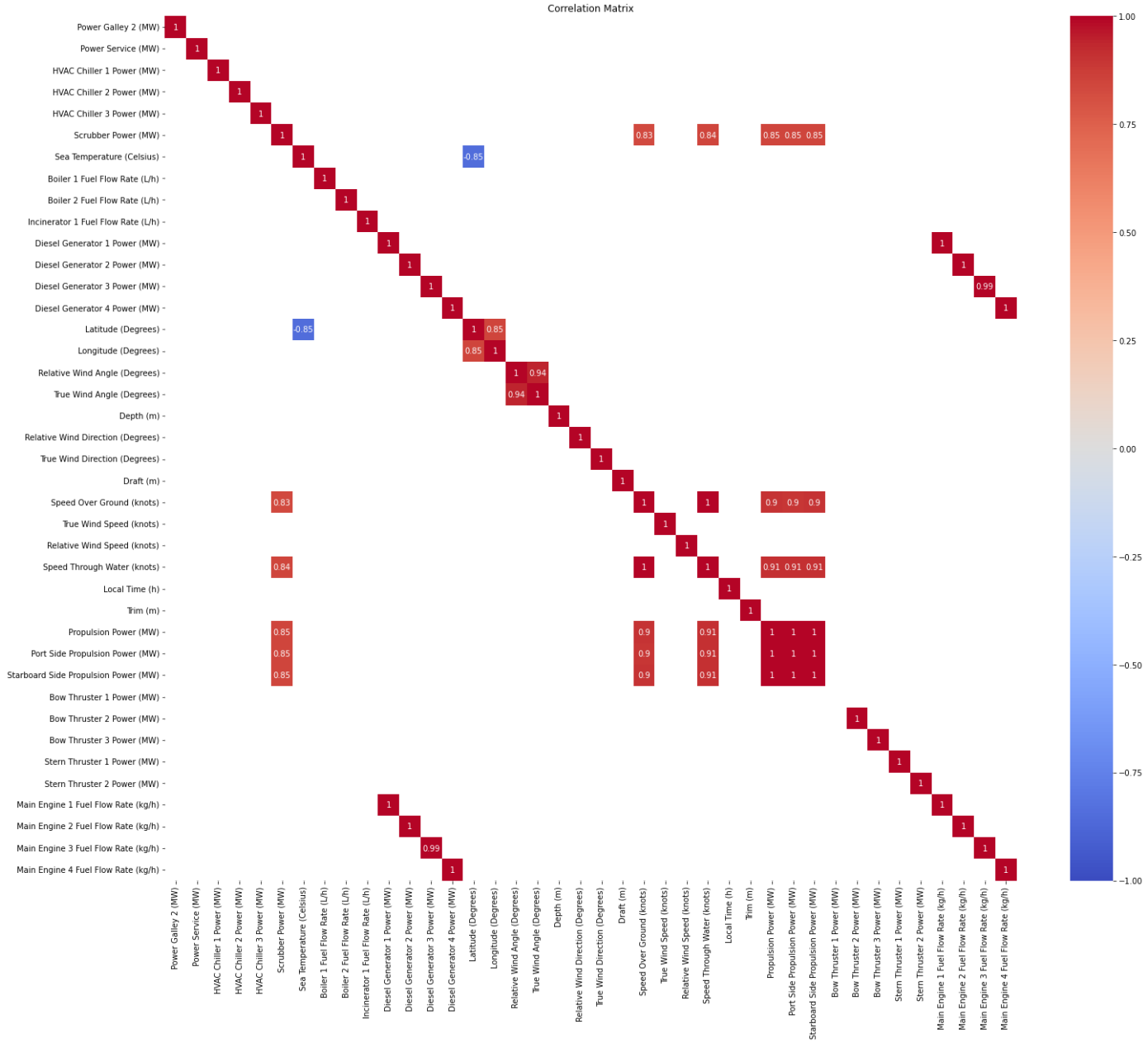
```
dfv.head()
```

Multi-Collinearity Check

```
# Correlation matrix
dfv_sub = dfv.iloc[:,3:]
corr = dfv_sub.corr()
```

```
high_corr = []
for i in corr.columns:
    high_corr.append(corr[(corr[i] > 0.8) | (corr[i] < -0.8)][i])
print(f'The highly correlated columns are:{high_corr}')
```

```
# Plot the correlation matrix
fig, ax = plt.subplots(figsize=(24,20))
sns.heatmap(data = corr[(corr > 0.8) | (corr < -0.8)], vmin=-1,vmax=1, cmap='coolwarm', ax = ax, annot= True)
ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
ax.set_title('Correlation Matrix')
plt.show()
```



Comments (Vessel 1):

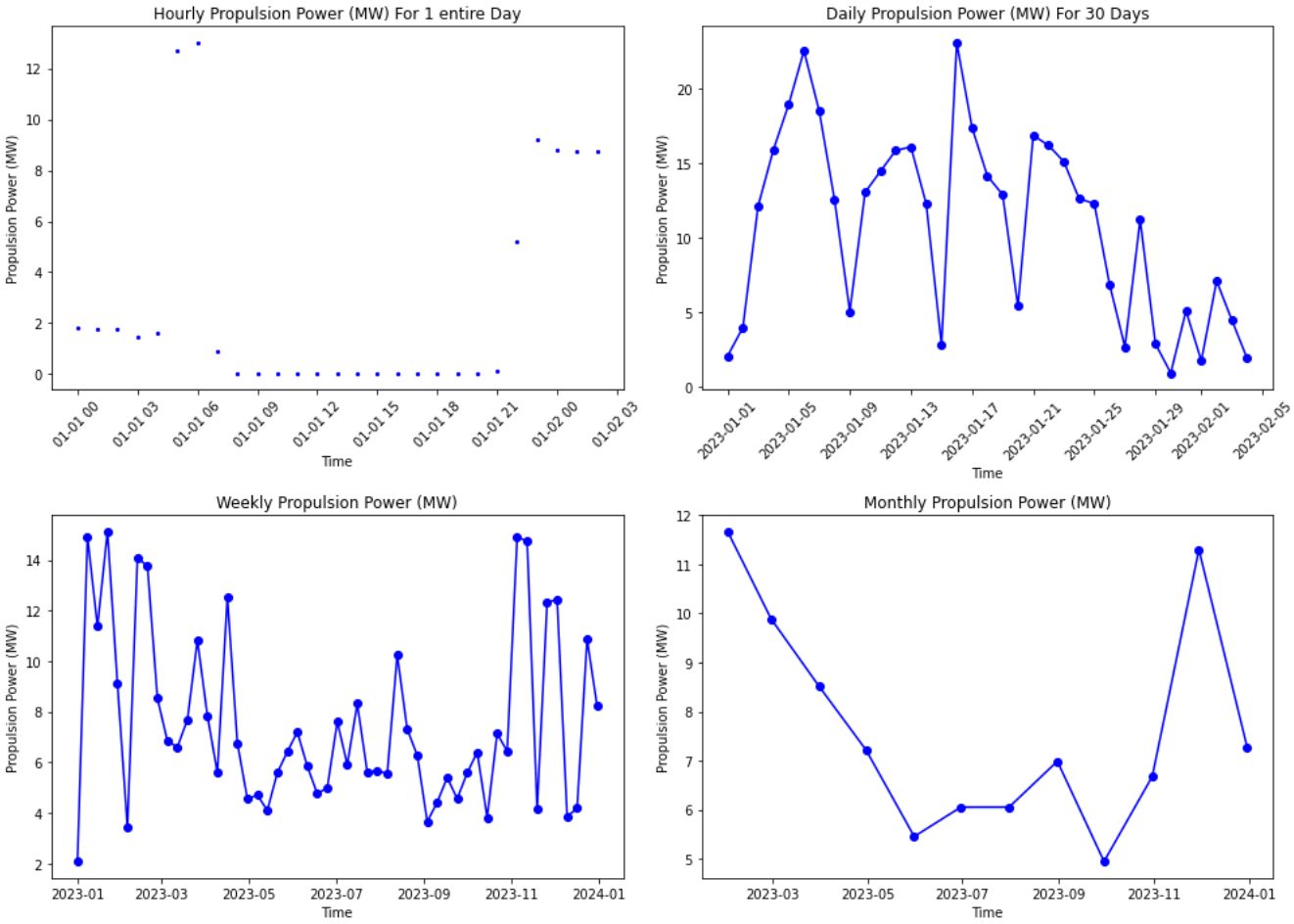
1. Diesel Generators' Power are highly correlated to corresponding Main Engine's Fuel flow rate
2. Propulsion power is a linear combination of Port Side Propulsion Power and Starboard Side Propulsion Power
3. Speed through water and Speed over ground are positively correlated to the Propulsion Power
4. Sea water temperature reducing as the latitude is increasing completely makes sense. This is because, temperatures are lower as one moves towards the poles

Trend and seasonality analysis

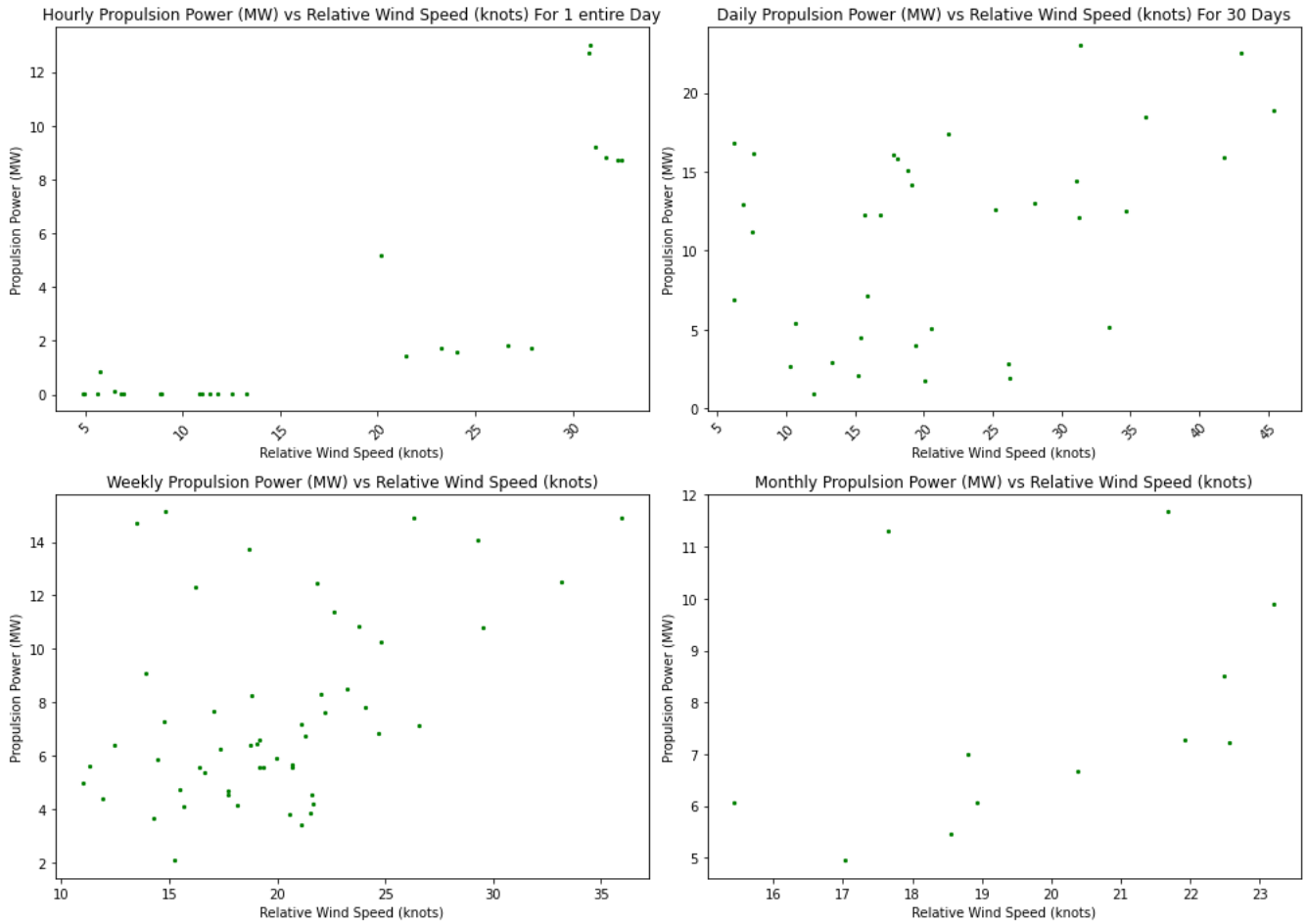
The outliers and skewness in the data are kept for analysis as this is recorded first hand. For forecasting the data will be standardized but now the analysis is done as is

```
# Resampling the data for hourly, daily, weekly and seasonal trends
hourly_df, daily_df, weekly_df, monthly_df = resample(dfv)
```

```
# Pick a feature from the list above to visualize the trend over
trend_plot(hourly_df, daily_df, weekly_df, monthly_df, 'Propulsion Power (MW)')
```




```
# Pick 2 features from the above list to visualize their relationship with each other over time
pair_plot(hourly_df, daily_df, weekly_df, monthly_df, 'Propulsion Power (MW)', 'Relative Wind Speed (knots)')
```



KPI Generation and Analysis

3 KPIs are generated based on domain knowledge and research (check the Energy Flow Diagram I built on the README):

1. Fuel Consumption per nautical mile (A measure of fuel efficiency)
2. Total Power Consumed
3. Power Specific Fuel Consumption

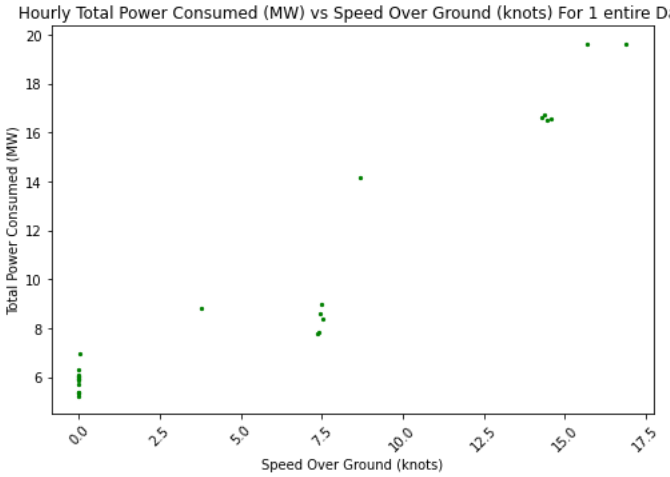
Method

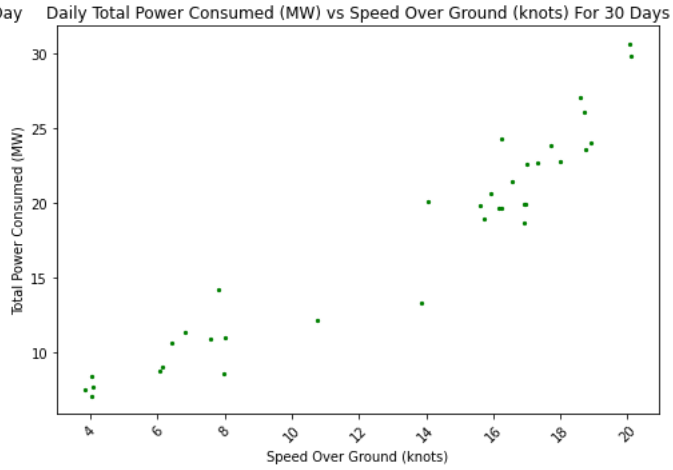
- Fuel Consumption per nautical mile = Sum of Engine Fuel Flow / Speed through water. It is capped at a maximum of 5000, because it goes to infinity when speed through is very close to 0. This happens while docking and anchoring
- Total Power Consumed = Sum power consumed by Power Galleys, HVAC Chillers, Power Service, Scrubber, Propulsion and Thrusters
- Power Specific Fuel Consumption = Fuel flow in engines, boilers and incinerator/ Total power consumed. Similar to the very commonly used fuel efficiency or brake specific fuel consumption (BSFC). It is used to assess the efficiency of any engine that burns fuel and produces rotational power, typically internal combustion engines. Assuming Fuel in Boiler and Incinerator to be Diesel and the density to be 0.85 kg/litre, for mass flow calculation.

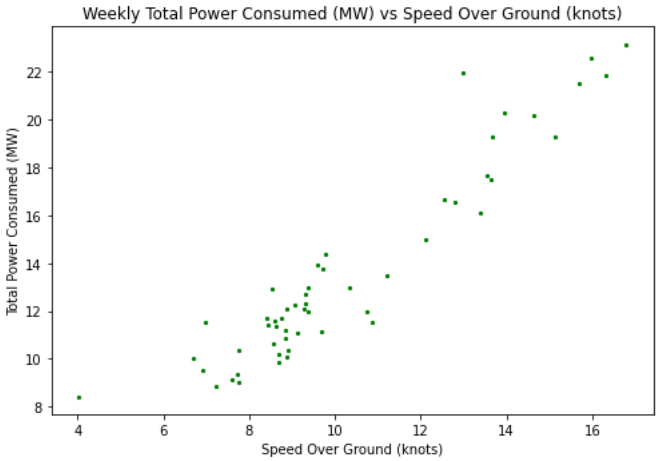
```
h_kpi['Fuel Consumption per nautical mile'].describe()
```

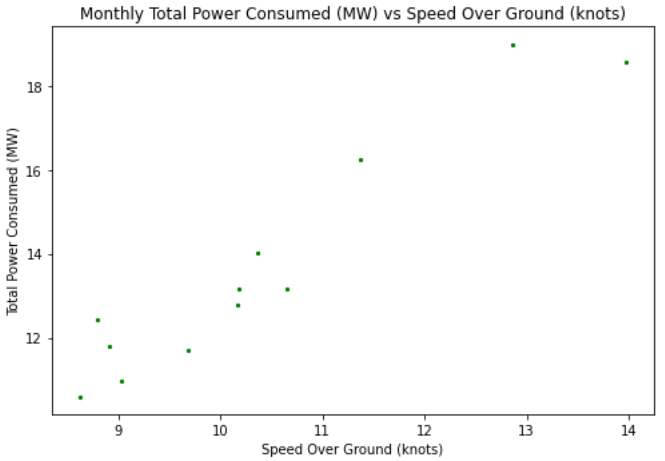
```
# Compute KPIS for daily, weekly and monthly granular data
d_kpi = compute_kpis(daily_df)
w_kpi = compute_kpis(weekly_df)
m_kpi = compute_kpis(monthly_df)
```

```
pair_plot(h_kpi, d_kpi, w_kpi, m_kpi, 'Total Power Consumed (MW)', 'Speed Over Ground (knots)')
```

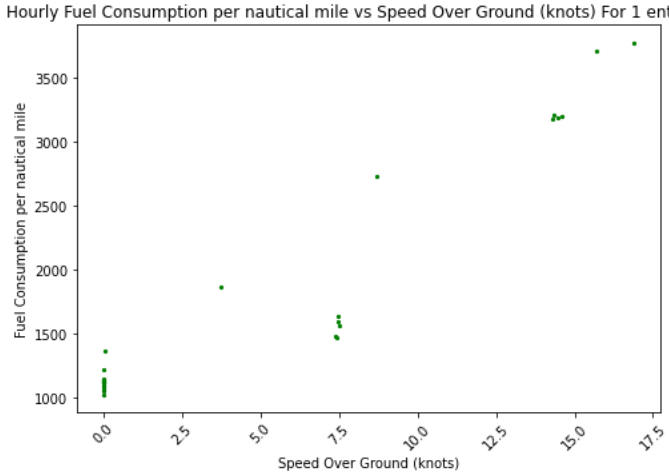


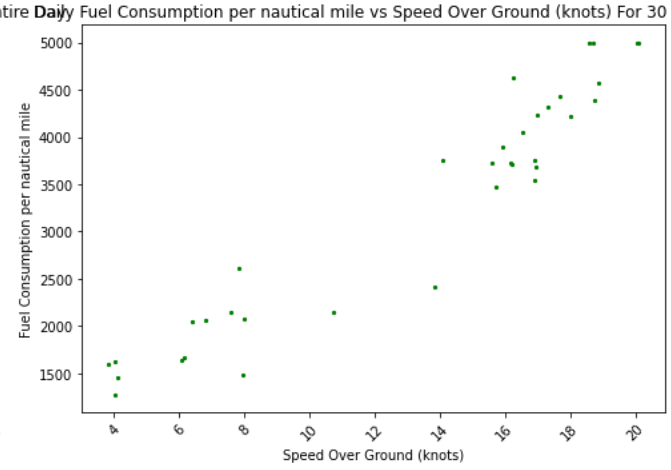


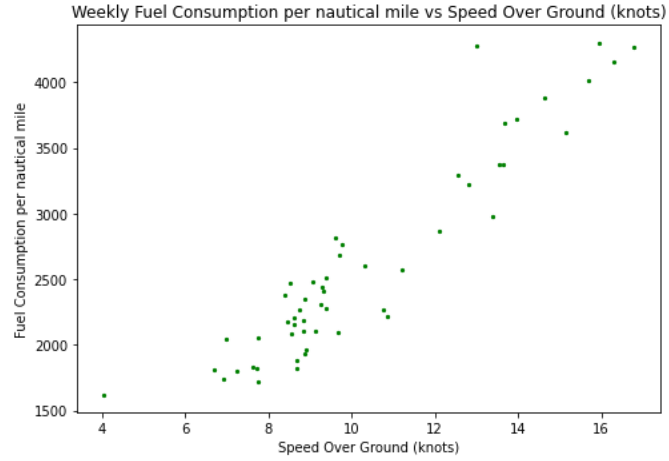


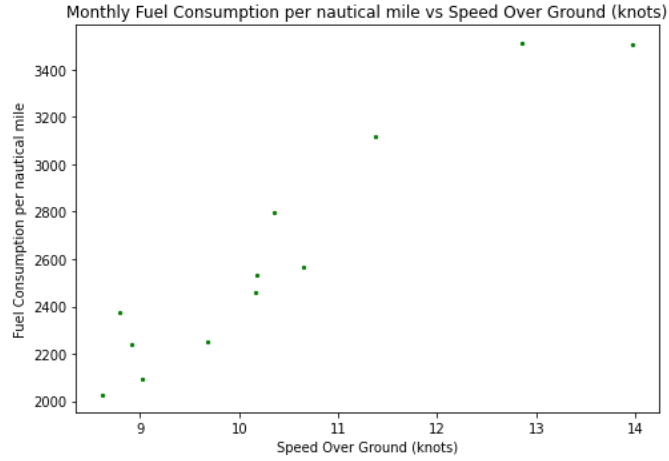



```
pair_plot(h_kpi, d_kpi, w_kpi, m_kpi, 'Fuel Consumption per nautical mile', 'Speed Over Ground (knots)')
```

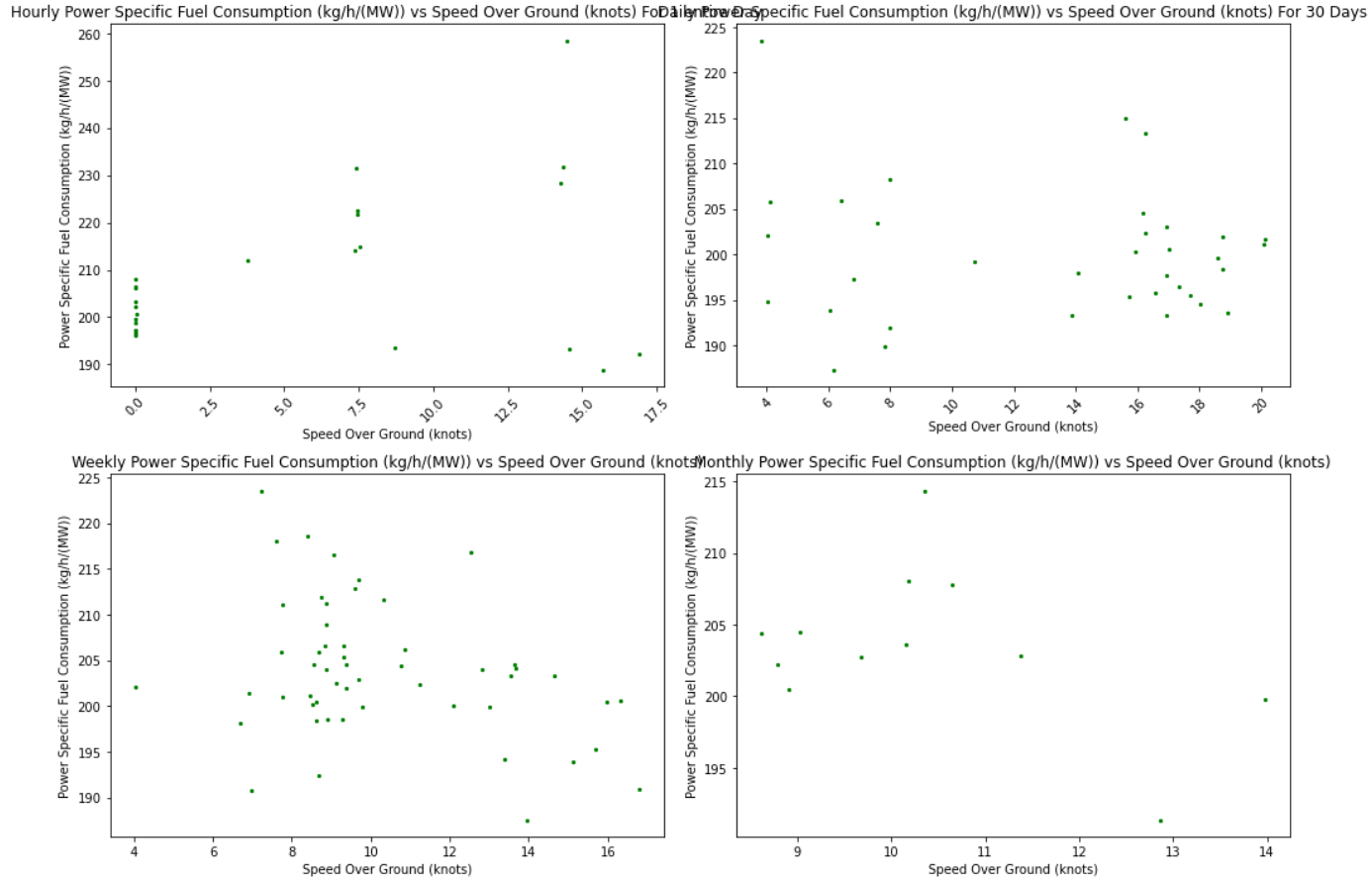








```
pair_plot(h_kpi, d_kpi, w_kpi, m_kpi, 'Power Specific Fuel Consumption (kg/h/(MW))', 'Speed Over Ground (knots)')
```



Comments(Vessel 1):

1. KPIs vs Sea Temperature

- On the long-term front, with seasonality in play: we can observe from the monthly chart that for warmer conditions of water, more power is consumed.
- This totally makes sense as the water would be less dense and the ship would have to work generate thrust to move through it
- The effect is not fully observed in the Fuel Efficiency chart as lower density also means lower resistance to move through water.

2. KPIs vs Speed over Ground

- Total Power Required increases with Speed over Ground
- Fuel Consumption per nautical mile has two clusters where there is not much increase in Fuel required per nautical mile (between 6&8, 12&17)
- Coming to Power Specific Fuel Consumption, the optimal speeds again lie between 12 and 17 where ther Fuel required to produce a certain power is flat and considerably low

Comments (Vessel 1)

- Based on the scatter plots, the optimal speed of operation are betweent 7.5-8.5 and 15-17.5, where there is a horizontal spread of points: the fuel required to operate in these range of speeds seem to be the same and not fluctuate a lot

Time-Series Forecasts

1. I first remove the columns not necessary for analysis like 'End Time' and 'Vessel Name'
2. Further, from the above analysis it is also determined that some columns are very highly correlated. If two columns are correlated at > 0.9, only one is kept for the analysis
3. If the distribution of a column is very narrow, it doesn't offer any variation as a feature. Hence we remove that also from the analysis
4. None of the features used to generate the KPI can be used for forecasting as it will be linearly dependant by inherent nature
5. Therefores, in the SARIMAX Seasonal modeling and forecast, I take an approach to try and forecast the total power from external factors that can be pre-planned before the cruise starts based on weather forecasts and other requirements
6. Finally, this is concluded with some simple hyperparameter tuning for better AIC value (Penalizing method for addition of a new feature to the model) to get the best performing forecast

Hourly will be too granular and computationally expensive. Monthly analysis would be sparse and not so informative since we only have 1 year data

External factors that can be pre-planned before the cruise starts

['Sea Temperature (Celsius)', 'Latitude (Degrees)', 'Longitude (Degrees)', 'Relative Wind Angle (Degrees)', 'True Wind Angle (Degrees)', 'Depth (m)', 'Relative Wind Direction (Degrees)', 'True Wind Direction (Degrees)', 'Draft (m)', 'Speed Over Ground (knots)', 'True Wind Speed (knots)', 'Relative Wind Speed (knots)', 'Local Time (h)',

'Trim (m)', 'Total Power Consumed (MW)']

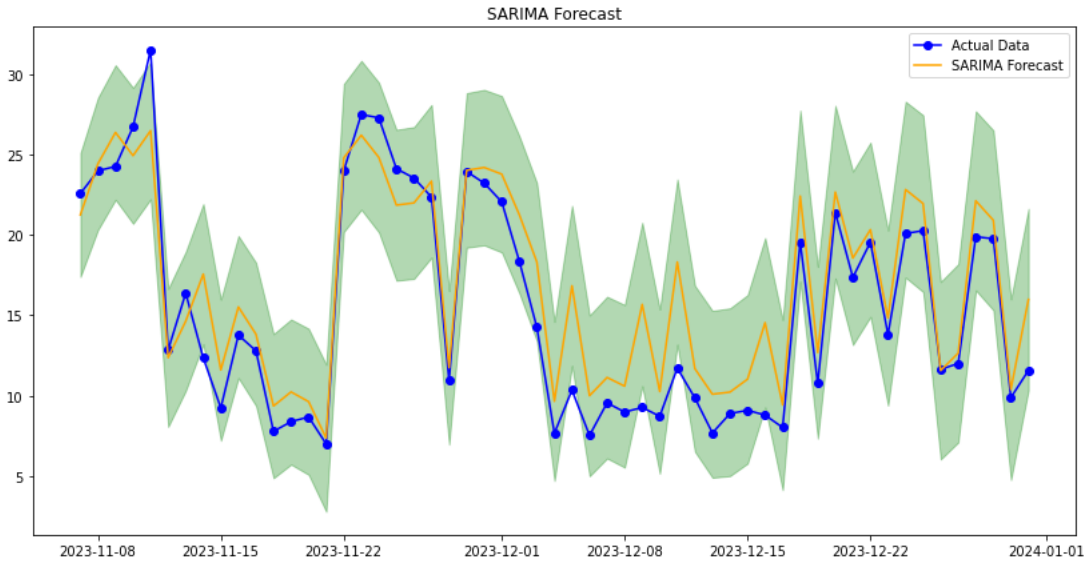
```
# Hourly forecast, the grid search for hyperparameters takes a long time to run, processor limitations
h_forecast = h_kpi[['Sea Temperature (Celsius)', 'Latitude (Degrees)', 'Longitude (Degrees)', 'Relative Wind Angle (Degrees)',
'True Wind Angle (Degrees)', 'Depth (m)', 'Relative Wind Direction (Degrees)', 'True Wind Direction (Degrees)', 'Draft (m)',
       'Speed Over Ground (knots)', 'True Wind Speed (knots)', 'Relative Wind Speed (knots)', 'Local Time (h)', 'Trim (m)',
'Total Power Consumed (MW)']]
```

```
# Daily forecast
d_forecast = d_kpi[['Sea Temperature (Celsius)', 'Latitude (Degrees)', 'Longitude (Degrees)', 'Relative Wind Angle (Degrees)',
'True Wind Angle (Degrees)', 'Depth (m)', 'Relative Wind Direction (Degrees)', 'True Wind Direction (Degrees)', 'Draft (m)',
       'Speed Over Ground (knots)', 'True Wind Speed (knots)', 'Relative Wind Speed (knots)', 'Local Time (h)', 'Trim (m)',
'Total Power Consumed (MW)']]
```

```
sarimax_forecast(d_forecast, 'Total Power Consumed (MW)', 7) #7 day forecast
```

```
Parameter combinations for SARIMA...[(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)] x
[(0, 0, 0, 7), (0, 0, 1, 7), (0, 1, 0, 7), (0, 1, 1, 7), (1, 0, 0, 7), (1, 0, 1, 7), (1, 1, 0, 7), (1, 1, 1, 7)]
SARIMA(0, 0, 0)x(0, 0, 0, 7) - AIC:2524.534106193414
SARIMA(0, 0, 0)x(0, 0, 1, 7) - AIC:2203.8287094025195
SARIMA(0, 0, 0)x(0, 1, 0, 7) - AIC:1495.6689137248247
SARIMA(0, 0, 0)x(0, 1, 1, 7) - AIC:1381.6930851306543
SARIMA(0, 0, 0)x(1, 0, 0, 7) - AIC:1554.3164392783694
SARIMA(0, 0, 0)x(1, 0, 1, 7) - AIC:1440.4675582552318
SARIMA(0, 0, 0)x(1, 1, 0, 7) - AIC:1440.8541347430978
SARIMA(0, 0, 0)x(1, 1, 1, 7) - AIC:1381.5532990178663
SARIMA(0, 0, 1)x(0, 0, 0, 7) - AIC:2163.522797180879
SARIMA(0, 0, 1)x(0, 0, 1, 7) - AIC:1953.7048368439919
SARIMA(0, 0, 1)x(0, 1, 0, 7) - AIC:1480.748203983198
SARIMA(0, 0, 1)x(0, 1, 1, 7) - AIC:1356.532035470119
SARIMA(0, 0, 1)x(1, 0, 0, 7) - AIC:1533.6655203857724
SARIMA(0, 0, 1)x(1, 0, 1, 7) - AIC:1410.7956904684083
SARIMA(0, 0, 1)x(1, 1, 0, 7) - AIC:1423.6449679588536
SARIMA(0, 0, 1)x(1, 1, 1, 7) - AIC:1357.0080661219745
SARIMA(0, 1, 0)x(0, 0, 0, 7) - AIC:1417.1614173512103
SARIMA(0, 1, 0)x(0, 0, 1, 7) - AIC:1419.0744101440982
SARIMA(0, 1, 0)x(0, 1, 0, 7) - AIC:1578.1572093749292
SARIMA(0, 1, 0)x(0, 1, 1, 7) - AIC:1413.5585797519866
SARIMA(0, 1, 0)x(1, 0, 0, 7) - AIC:1419.0944385993396
SARIMA(0, 1, 0)x(1, 0, 1, 7) - AIC:1420.3892379002527
SARIMA(0, 1, 0)x(1, 1, 0, 7) - AIC:1518.9672706609258
SARIMA(0, 1, 0)x(1, 1, 1, 7) - AIC:1415.5420908756155
SARIMA(0, 1, 1)x(0, 0, 0, 7) - AIC:1337.99305600544
SARIMA(0, 1, 1)x(0, 0, 1, 7) - AIC:1339.6949213284652
SARIMA(0, 1, 1)x(0, 1, 0, 7) - AIC:1498.5847534599445
SARIMA(0, 1, 1)x(0, 1, 1, 7) - AIC:1339.4247999572608
SARIMA(0, 1, 1)x(1, 0, 0, 7) - AIC:1339.7699368738986
SARIMA(0, 1, 1)x(1, 0, 1, 7) - AIC:1339.2871913507195
SARIMA(0, 1, 1)x(1, 1, 0, 7) - AIC:1441.8920102570764
SARIMA(0, 1, 1)x(1, 1, 1, 7) - AIC:1341.2628219283417
SARIMA(1, 0, 0)x(0, 0, 0, 7) - AIC:1425.1160604183726
SARIMA(1, 0, 0)x(0, 0, 1, 7) - AIC:1426.9053862956748
SARIMA(1, 0, 0)x(0, 1, 0, 7) - AIC:1475.6631209066668
SARIMA(1, 0, 0)x(0, 1, 1, 7) - AIC:1343.1015926486082
SARIMA(1, 0, 0)x(1, 0, 0, 7) - AIC:1426.9623980342653
SARIMA(1, 0, 0)x(1, 0, 1, 7) - AIC:1394.4765352557101
SARIMA(1, 0, 0)x(1, 1, 0, 7) - AIC:1419.0929052089123
SARIMA(1, 0, 0)x(1, 1, 1, 7) - AIC:1343.9278744956437
SARIMA(1, 0, 1)x(0, 0, 0, 7) - AIC:1348.350567114666
SARIMA(1, 0, 1)x(0, 0, 1, 7) - AIC:1350.0651408508406
SARIMA(1, 0, 1)x(0, 1, 0, 7) - AIC:1476.5414219993563
SARIMA(1, 0, 1)x(0, 1, 1, 7) - AIC:1336.229878608054
SARIMA(1, 0, 1)x(1, 0, 0, 7) - AIC:1350.2525339597723
SARIMA(1, 0, 1)x(1, 0, 1, 7) - AIC:1352.3905059556037
SARIMA(1, 0, 1)x(1, 1, 0, 7) - AIC:1420.7313919852566
SARIMA(1, 0, 1)x(1, 1, 1, 7) - AIC:1337.8789967756884
SARIMA(1, 1, 0)x(0, 0, 0, 7) - AIC:1378.174872938987
SARIMA(1, 1, 0)x(0, 0, 1, 7) - AIC:1380.1733105012263
SARIMA(1, 1, 0)x(0, 1, 0, 7) - AIC:1541.5180719156613
SARIMA(1, 1, 0)x(0, 1, 1, 7) - AIC:1377.5584622385186
```

```
SARIMA(1, 1, 0)x(1, 0, 0, 7) - AIC:1380.173780859042
SARIMA(1, 1, 0)x(1, 0, 1, 7) - AIC:1379.5828648327788
SARIMA(1, 1, 0)x(1, 1, 0, 7) - AIC:1488.0021531164077
SARIMA(1, 1, 0)x(1, 1, 1, 7) - AIC:1379.5694659850183
SARIMA(1, 1, 1)x(0, 0, 0, 7) - AIC:1330.37243216749
SARIMA(1, 1, 1)x(0, 0, 1, 7) - AIC:1332.3376572590037
SARIMA(1, 1, 1)x(0, 1, 0, 7) - AIC:1478.7393210887928
SARIMA(1, 1, 1)x(0, 1, 1, 7) - AIC:1331.562339377681
SARIMA(1, 1, 1)x(1, 0, 0, 7) - AIC:1332.3450261667115
SARIMA(1, 1, 1)x(1, 0, 1, 7) - AIC:1333.5589045789138
SARIMA(1, 1, 1)x(1, 1, 0, 7) - AIC:1422.5167709527489
SARIMA(1, 1, 1)x(1, 1, 1, 7) - AIC:1333.5583192797858
Best SARIMA(1, 1, 1)x(0, 0, 0, 7) - AIC:1330.37243216749
RMSE:  2.616105733879915
```



- For the dataset on a daily level granularity, Total power consumption for the cruise ship can be forecasted within acceptable margin of errors. The 95% confidence range of forecast is also predicted. The true range almost always lies in the 95% confidence range of forecast.
- This is by using external factors only, without taking into account any operational component. Therefore it states that the external conditions can forecast the energy requirements for the ship, this can aid in early resource planning of cruises
- Best SARIMA(1, 1, 1)x(0, 0, 0, 7) - AIC:1330.37243216749
- RMSE: 2.616105733879915 (MW)

```
d_forecast_fuel = d_kpi[['Fuel Consumption per nautical mile', 'Total Power Consumed (MW)']]
```

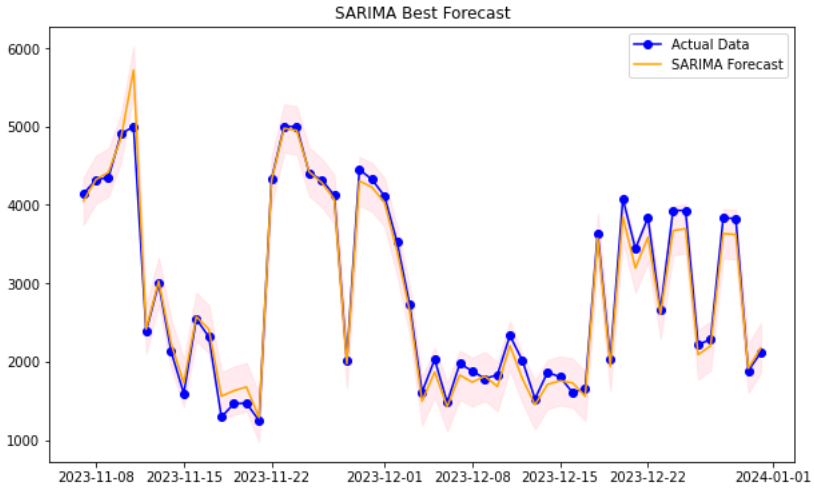
```
d_forecast_fuel.head()
```

```
sarimax_forecast(d_forecast_fuel, 'Fuel Consumption per nautical mile', 7)
```

- For the dataset on a daily level granularity, Fuel Consumption per nautical mile for the cruise ship can be forecasted just using the Total power consumption data within acceptable range of errors. The 95% confidence range of forecast is also predicted. The true range almost always lies in the 95% confidence range of forecast. The 95% range in this case is also narrow, that shows more certainity and less variance in predictions.
- This means, if an accurate enough forecast can be made for the power requirements, the forecast can in turn can be used for fuel requirements of the cruise ship. Therefore it states that the external conditions can forecast the fuel planning requirements for the ship, this can aid in early duel planning of cruises, with potential for entering trade positions.
- Best SARIMA(1, 1, 1)x(0, 1, 1, 7) - AIC:3924.477617510145
- RMSE: 161.68641766048893 (kg/h/knots)

```
best_forecast(d_forecast_fuel, 'Fuel Consumption per nautical mile', [1,1,1],[0,1,1,7])
```

```
RMSE:  161.68641766048893
```

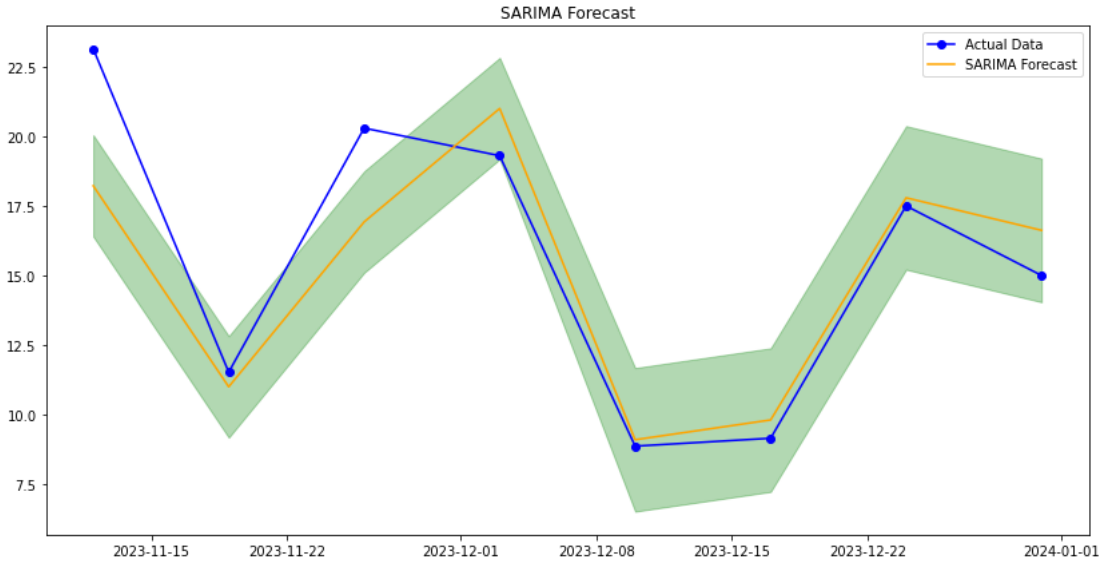



```
# Weekly Forecast model
w_forecast = w_kpi[['Sea Temperature (Celsius)', 'Latitude (Degrees)', 'Longitude (Degrees)', 'Relative Wind Angle (Degrees)',
'True Wind Angle (Degrees)', 'Depth (m)', 'Relative Wind Direction (Degrees)', 'True Wind Direction (Degrees)', 'Draft (m)',
       'Speed Over Ground (knots)', 'True Wind Speed (knots)', 'Relative Wind Speed (knots)', 'Local Time (h)', 'Trim (m)',
'Total Power Consumed (MW)']]
```

```
sarimax_forecast(w_forecast, 'Total Power Consumed (MW)', 4)
```

```
Parameter combinations for SARIMA...[(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)] x
[(0, 0, 0, 4), (0, 0, 1, 4), (0, 1, 0, 4), (0, 1, 1, 4), (1, 0, 0, 4), (1, 0, 1, 4), (1, 1, 0, 4), (1, 1, 1, 4)]
SARIMA(0, 0, 0)x(0, 0, 0, 4) - AIC:390.37517580742025
SARIMA(0, 0, 0)x(0, 0, 1, 4) - AIC:343.6143444133899
SARIMA(0, 0, 0)x(0, 1, 0, 4) - AIC:140.56432053018307
SARIMA(0, 0, 0)x(0, 1, 1, 4) - AIC:142.54787539924934
SARIMA(0, 0, 0)x(1, 0, 0, 4) - AIC:175.61986159649464
SARIMA(0, 0, 0)x(1, 0, 1, 4) - AIC:177.2582020204029
SARIMA(0, 0, 0)x(1, 1, 0, 4) - AIC:142.51175321804237
SARIMA(0, 0, 0)x(1, 1, 1, 4) - AIC:144.2954582877191
SARIMA(0, 0, 1)x(0, 0, 0, 4) - AIC:335.33682665284067
SARIMA(0, 0, 1)x(0, 0, 1, 4) - AIC:291.51265112477734
SARIMA(0, 0, 1)x(0, 1, 0, 4) - AIC:142.46372952879415
SARIMA(0, 0, 1)x(0, 1, 1, 4) - AIC:144.3305496829376
SARIMA(0, 0, 1)x(1, 0, 0, 4) - AIC:177.5382105907911
SARIMA(0, 0, 1)x(1, 0, 1, 4) - AIC:181.0619585260291
SARIMA(0, 0, 1)x(1, 1, 0, 4) - AIC:144.24962320007256
SARIMA(0, 0, 1)x(1, 1, 1, 4) - AIC:146.104453392421
SARIMA(0, 1, 0)x(0, 0, 0, 4) - AIC:159.55120681962225
SARIMA(0, 1, 0)x(0, 0, 1, 4) - AIC:151.51191936815928
SARIMA(0, 1, 0)x(0, 1, 0, 4) - AIC:153.09975542822258
SARIMA(0, 1, 0)x(0, 1, 1, 4) - AIC:154.84249639973893
SARIMA(0, 1, 0)x(1, 0, 0, 4) - AIC:157.64303801574505
SARIMA(0, 1, 0)x(1, 0, 1, 4) - AIC:151.02611861709582
SARIMA(0, 1, 0)x(1, 1, 0, 4) - AIC:154.44315858254305
SARIMA(0, 1, 0)x(1, 1, 1, 4) - AIC:156.39924414490042
SARIMA(0, 1, 1)x(0, 0, 0, 4) - AIC:148.68196483312835
SARIMA(0, 1, 1)x(0, 0, 1, 4) - AIC:148.84641427844224
SARIMA(0, 1, 1)x(0, 1, 0, 4) - AIC:143.8175494748058
SARIMA(0, 1, 1)x(0, 1, 1, 4) - AIC:145.75113249975794
SARIMA(0, 1, 1)x(1, 0, 0, 4) - AIC:147.38704538609198
SARIMA(0, 1, 1)x(1, 0, 1, 4) - AIC:149.07494786689318
SARIMA(0, 1, 1)x(1, 1, 0, 4) - AIC:145.65956817450564
SARIMA(0, 1, 1)x(1, 1, 1, 4) - AIC:147.3430413145698
SARIMA(1, 0, 0)x(0, 0, 0, 4) - AIC:169.80812970269645
SARIMA(1, 0, 0)x(0, 0, 1, 4) - AIC:166.351058067871
SARIMA(1, 0, 0)x(0, 1, 0, 4) - AIC:142.53460773891885
SARIMA(1, 0, 0)x(0, 1, 1, 4) - AIC:144.46405610495958
SARIMA(1, 0, 0)x(1, 0, 0, 4) - AIC:206.83288740380019
SARIMA(1, 0, 0)x(1, 0, 1, 4) - AIC:173.81113829489516
SARIMA(1, 0, 0)x(1, 1, 0, 4) - AIC:144.38366909957398
SARIMA(1, 0, 0)x(1, 1, 1, 4) - AIC:146.19436088956303
SARIMA(1, 0, 1)x(0, 0, 0, 4) - AIC:160.26290540391796
SARIMA(1, 0, 1)x(0, 0, 1, 4) - AIC:160.8320224447697
```

```
SARIMA(1, 0, 1)x(0, 1, 0, 4) - AIC:144.22699745912263
SARIMA(1, 0, 1)x(0, 1, 1, 4) - AIC:146.18911473291485
SARIMA(1, 0, 1)x(1, 0, 0, 4) - AIC:180.17950268679672
SARIMA(1, 0, 1)x(1, 0, 1, 4) - AIC:168.1647886181343
SARIMA(1, 0, 1)x(1, 1, 0, 4) - AIC:146.08362740222915
SARIMA(1, 0, 1)x(1, 1, 1, 4) - AIC:147.84771278377445
SARIMA(1, 1, 0)x(0, 0, 0, 4) - AIC:153.42918294311346
SARIMA(1, 1, 0)x(0, 0, 1, 4) - AIC:146.07658655823408
SARIMA(1, 1, 0)x(0, 1, 0, 4) - AIC:151.53813684379202
SARIMA(1, 1, 0)x(0, 1, 1, 4) - AIC:152.51776755379072
SARIMA(1, 1, 0)x(1, 0, 0, 4) - AIC:147.6404446398443
SARIMA(1, 1, 0)x(1, 0, 1, 4) - AIC:149.41518648540608
SARIMA(1, 1, 0)x(1, 1, 0, 4) - AIC:145.5132964660383
SARIMA(1, 1, 0)x(1, 1, 1, 4) - AIC:146.98759703752125
SARIMA(1, 1, 1)x(0, 0, 0, 4) - AIC:150.467476574397
SARIMA(1, 1, 1)x(0, 0, 1, 4) - AIC:148.0516440196325
SARIMA(1, 1, 1)x(0, 1, 0, 4) - AIC:145.7465095280253
SARIMA(1, 1, 1)x(0, 1, 1, 4) - AIC:147.56954916885198
SARIMA(1, 1, 1)x(1, 0, 0, 4) - AIC:148.36455068247955
SARIMA(1, 1, 1)x(1, 0, 1, 4) - AIC:149.12707402803065
SARIMA(1, 1, 1)x(1, 1, 0, 4) - AIC:147.4286713129202
SARIMA(1, 1, 1)x(1, 1, 1, 4) - AIC:148.81140272069416
Best SARIMA(0, 0, 0)x(0, 1, 0, 4) - AIC:140.56432053018307
RMSE:  2.291589529625523
```



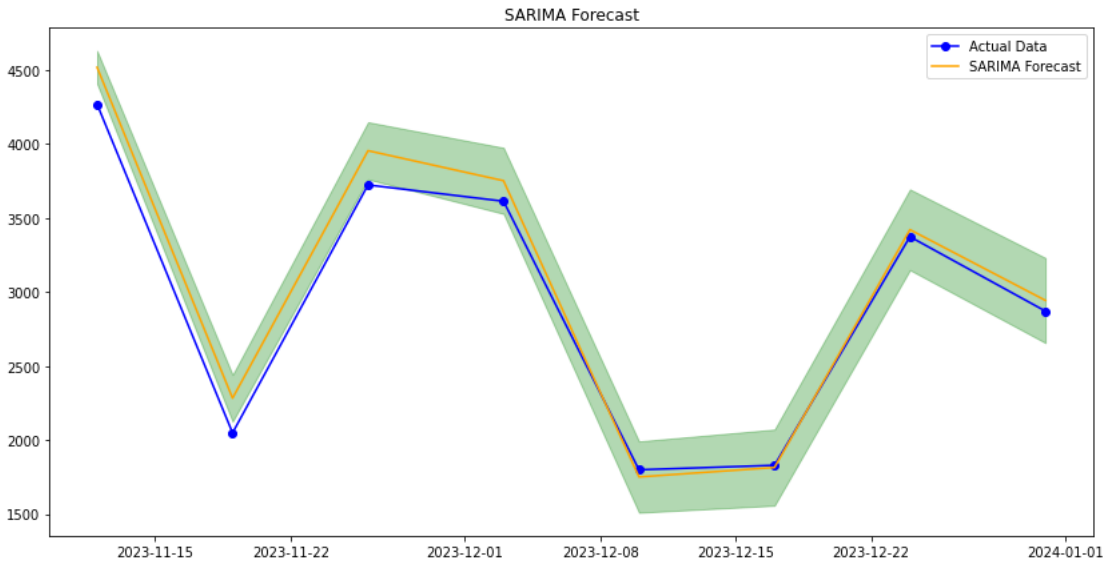
The same deductions as daily forecasts apply for the weekly analyses too: Best SARIMA(0, 0, 0)x(0, 1, 0, 4) - AIC:140.56432053018307 RMSE: 2.291589529625523

```
w_forecast_fuel = w_kpi[['Fuel Consumption per nautical mile', 'Total Power Consumed (MW)']]
```

```
sarimax_forecast(w_forecast_fuel, 'Fuel Consumption per nautical mile', 4)  # Weekly forecast 4: weeks constituting a month
```

```
Parameter combinations for SARIMA...[(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)] x
[(0, 0, 0, 4), (0, 0, 1, 4), (0, 1, 0, 4), (0, 1, 1, 4), (1, 0, 0, 4), (1, 0, 1, 4), (1, 1, 0, 4), (1, 1, 1, 4)]
SARIMA(0, 0, 0)x(0, 0, 0, 4) - AIC:837.4418970354757
SARIMA(0, 0, 0)x(0, 0, 1, 4) - AIC:791.245659308201
SARIMA(0, 0, 0)x(0, 1, 0, 4) - AIC:494.6302944232595
SARIMA(0, 0, 0)x(0, 1, 1, 4) - AIC:473.83139841524354
SARIMA(0, 0, 0)x(1, 0, 0, 4) - AIC:570.6733132554124
SARIMA(0, 0, 0)x(1, 0, 1, 4) - AIC:555.1275227241101
SARIMA(0, 0, 0)x(1, 1, 0, 4) - AIC:486.3414879672588
SARIMA(0, 0, 0)x(1, 1, 1, 4) - AIC:475.58416789044077
SARIMA(0, 0, 1)x(0, 0, 0, 4) - AIC:781.921644449832
SARIMA(0, 0, 1)x(0, 0, 1, 4) - AIC:734.5467249752637
SARIMA(0, 0, 1)x(0, 1, 0, 4) - AIC:483.073104098533
SARIMA(0, 0, 1)x(0, 1, 1, 4) - AIC:459.71926699140585
SARIMA(0, 0, 1)x(1, 0, 0, 4) - AIC:830.0152488027388
SARIMA(0, 0, 1)x(1, 0, 1, 4) - AIC:731.5873180926752
SARIMA(0, 0, 1)x(1, 1, 0, 4) - AIC:473.99478587833363
```

```
SARIMA(0, 0, 1)x(1, 1, 1, 4) - AIC:460.7883510367483
SARIMA(0, 1, 0)x(0, 0, 0, 4) - AIC:482.04858799980576
SARIMA(0, 1, 0)x(0, 0, 1, 4) - AIC:475.9669503040916
SARIMA(0, 1, 0)x(0, 1, 0, 4) - AIC:478.3309951435947
SARIMA(0, 1, 0)x(0, 1, 1, 4) - AIC:453.1789347646319
SARIMA(0, 1, 0)x(1, 0, 0, 4) - AIC:478.9721713275772
SARIMA(0, 1, 0)x(1, 0, 1, 4) - AIC:475.7283216780236
SARIMA(0, 1, 0)x(1, 1, 0, 4) - AIC:463.959286972216
SARIMA(0, 1, 0)x(1, 1, 1, 4) - AIC:452.4464831963399
SARIMA(0, 1, 1)x(0, 0, 0, 4) - AIC:483.9958221027617
SARIMA(0, 1, 1)x(0, 0, 1, 4) - AIC:477.9567376882776
SARIMA(0, 1, 1)x(0, 1, 0, 4) - AIC:480.2590614421333
SARIMA(0, 1, 1)x(0, 1, 1, 4) - AIC:455.1356842369234
SARIMA(0, 1, 1)x(1, 0, 0, 4) - AIC:480.88687245030803
SARIMA(0, 1, 1)x(1, 0, 1, 4) - AIC:477.6878516774099
SARIMA(0, 1, 1)x(1, 1, 0, 4) - AIC:465.26120511643916
SARIMA(0, 1, 1)x(1, 1, 1, 4) - AIC:454.38932790087006
SARIMA(1, 0, 0)x(0, 0, 0, 4) - AIC:502.5876233489073
SARIMA(1, 0, 0)x(0, 0, 1, 4) - AIC:496.53993748116574
SARIMA(1, 0, 0)x(0, 1, 0, 4) - AIC:482.82771984367037
SARIMA(1, 0, 0)x(0, 1, 1, 4) - AIC:457.6802371968709
SARIMA(1, 0, 0)x(1, 0, 0, 4) - AIC:500.7533661058035
SARIMA(1, 0, 0)x(1, 0, 1, 4) - AIC:506.6138253024592
SARIMA(1, 0, 0)x(1, 1, 0, 4) - AIC:469.43026103376707
SARIMA(1, 0, 0)x(1, 1, 1, 4) - AIC:457.6655981104512
SARIMA(1, 0, 1)x(0, 0, 0, 4) - AIC:504.5382551644803
SARIMA(1, 0, 1)x(0, 0, 1, 4) - AIC:498.53192930848843
SARIMA(1, 0, 1)x(0, 1, 0, 4) - AIC:483.58056541730315
SARIMA(1, 0, 1)x(0, 1, 1, 4) - AIC:458.78125421044865
SARIMA(1, 0, 1)x(1, 0, 0, 4) - AIC:551.7939128435235
SARIMA(1, 0, 1)x(1, 0, 1, 4) - AIC:508.54975243228034
SARIMA(1, 0, 1)x(1, 1, 0, 4) - AIC:471.3921299976956
SARIMA(1, 0, 1)x(1, 1, 1, 4) - AIC:458.9082223762439
SARIMA(1, 1, 0)x(0, 0, 0, 4) - AIC:484.00667819046777
SARIMA(1, 1, 0)x(0, 0, 1, 4) - AIC:477.96256023506163
SARIMA(1, 1, 0)x(0, 1, 0, 4) - AIC:480.26238503126115
SARIMA(1, 1, 0)x(0, 1, 1, 4) - AIC:455.1437497740863
SARIMA(1, 1, 0)x(1, 0, 0, 4) - AIC:480.9181791848227
SARIMA(1, 1, 0)x(1, 0, 1, 4) - AIC:477.7119746456858
SARIMA(1, 1, 0)x(1, 1, 0, 4) - AIC:465.23198917969614
SARIMA(1, 1, 0)x(1, 1, 1, 4) - AIC:454.4078917703319
SARIMA(1, 1, 1)x(0, 0, 0, 4) - AIC:479.1785598092354
SARIMA(1, 1, 1)x(0, 0, 1, 4) - AIC:474.9334586617688
SARIMA(1, 1, 1)x(0, 1, 0, 4) - AIC:475.48088067884885
SARIMA(1, 1, 1)x(0, 1, 1, 4) - AIC:453.49343785010194
SARIMA(1, 1, 1)x(1, 0, 0, 4) - AIC:477.5257609496002
SARIMA(1, 1, 1)x(1, 0, 1, 4) - AIC:474.80345961515565
SARIMA(1, 1, 1)x(1, 1, 0, 4) - AIC:463.7639186359571
SARIMA(1, 1, 1)x(1, 1, 1, 4) - AIC:453.71225742479555
Best SARIMA(0, 1, 0)x(1, 1, 1, 4) - AIC:452.4464831963399
RMSE:  158.0555408961258
```



Best SARIMA(0, 1, 0)x(1, 1, 1, 4) - AIC:452.4464831963399 RMSE: 158.0555408961258

```
weekly_df.reset_index(inplace=True)
weekly_df.head()
```

```
hourly_df.to_csv("hourly.csv")
```

```
# Generate column Table
table = '| Column Name |\n|-------------|\n'
for column in weekly_df.columns:
    table += f'| {column} |\n'
print(table)
```