In this notebook, we are gonna see if there are cases when one engine is off and the another one is on. We need this information because the one that is off has some issues that should be checked.

In [1]:

```
import pandas as pd

# Python standard library imports
import time

# Third-party imports for database connection and data manipulation
from sqlalchemy import create_engine
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as stats
import pandas as pd
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.covariance import EllipticEnvelope
from sklearn.svm import OneClassSVM
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
# Database connection parameters
dbname = 'train_weather'
user = 'narmina'
password = 'narmina7877'
host = 'localhost'  # localhost or the server address
port = '5433'  # default PostgreSQL port is 5432

# Establish a connection to the database
connection_str = f"postgresql://{user}:{password}@{host}:{port}/{dbname}"
engine = create_engine(connection_str)
```

In [3]:

```
# Define the query with placeholders for parameters
query1 = f"""
SELECT *, 'one_engine_off' AS outlier_type
FROM vehicle_data_enriched
where ("rs_e_rpm_pc1"=0 and "rs_e_rpm_pc2"!=0) or ("rs_e_rpm_pc2"=0 and "rs_
"""

# Start timing
start_time = time.time()

# Execute the query and fetch the data into a DataFrame
df1 = pd.read_sql_query(query1, engine)

# End timing
end_time = time.time()
print(f"Query took {end_time - start_time} seconds to run.")
# Close the database connection
engine.dispose()
```

```
Query took 0.2780911922454834 seconds to run.
```

In [4]:

```
print(df1)
```

```
       mapped_veh_id      timestamps_UTC        lat       lon  \
0              194.0  2023-08-24 20:34:22  51.035433  3.711526
1              194.0  2023-08-24 20:47:53  51.035310  3.712176
2              134.0  2023-08-01 17:52:34  51.205410  5.249654
3              187.0  2023-08-01 00:02:46  51.013895  3.779681
4              142.0  2023-08-01 12:48:50  50.936567  5.306654
...              ...                  ...        ...       ...
10669          172.0  2023-09-09 07:23:36  50.935316  5.310786
10670          131.0  2023-09-09 18:51:49  50.107995  4.500648
10671          131.0  2023-09-09 21:21:55  50.065206  4.496489
10672          172.0  2023-09-09 04:31:43  50.935317  5.310762
10673          172.0  2023-09-09 06:24:40  50.935453  5.310185

       RS_E_InAirTemp_PC1  RS_E_InAirTemp_PC2  RS_E_OilPress_PC1  \
0                     0.0                41.0                0.0
1                     0.0                38.0                0.0
2                    31.0                22.0                3.0
3                    60.0                33.0                0.0
4                    34.0                33.0              255.0
...                   ...                 ...                ...
10669                31.0                33.0                0.0
10670                53.0                62.0              690.0
10671                45.0                53.0              690.0
10672                33.0                27.0                0.0
10673                31.0                29.0                0.0

       RS_E_OilPress_PC2  rs_e_rpm_pc1  rs_e_rpm_pc2  ...  RS_T_OilTemp_PC2
\
0             210.000000           0.0    802.000000  ...              82.0
1             213.333333           0.0    799.666667  ...              82.0
2             224.000000           0.0    797.000000  ...              79.0
3             282.000000           0.0    796.000000  ...              83.0
4               3.000000         851.0      0.000000  ...              25.0
...                  ...           ...           ...  ...               ...
10669         231.000000           0.0    798.000000  ...              80.0
10670         690.000000        1395.0      0.000000  ...              71.0
10671         690.000000         823.0      0.000000  ...              54.0
10672         241.000000           0.0    799.000000  ...              79.0
10673         234.000000           0.0    794.000000  ...              79.0

          timestamps_floor  nearest_point_id        Lat       Lon  \
0      2023-08-24 20:00:00               208  51.035433  3.711526
1      2023-08-24 20:00:00               208  51.035310  3.712176
2      2023-08-01 17:00:00               145  51.205410  5.249654
3      2023-08-01 00:00:00               208  51.013895  3.779681
4      2023-08-01 12:00:00               262  50.936567  5.306654
...                    ...               ...        ...       ...
10669  2023-09-09 07:00:00               262  50.935316  5.310786
10670  2023-09-09 18:00:00               566  50.107995  4.500648
10671  2023-09-09 21:00:00               566  50.065206  4.496489
10672  2023-09-09 04:00:00               262  50.935317  5.310762
10673  2023-09-09 06:00:00               262  50.935453  5.310185

                      Time  Temperature  Humidity Rain    outlier_type
0      2023-08-24 20:00:00         21.2      85.0  0.0  one_engine_off
1      2023-08-24 20:00:00         21.2      85.0  0.0  one_engine_off
2      2023-08-01 17:00:00         18.4      65.0  0.0  one_engine_off
3      2023-08-01 00:00:00         15.8      91.0  0.0  one_engine_off
4      2023-08-01 12:00:00         19.7      67.0  0.0  one_engine_off
...                    ...          ...       ...  ...             ...
10669  2023-09-09 07:00:00         19.7      90.0  0.0  one_engine_off
10670  2023-09-09 18:00:00         24.9      72.0  0.0  one_engine_off
10671  2023-09-09 21:00:00         21.7      80.0  0.0  one_engine_off
10672  2023-09-09 04:00:00         15.8      99.0  0.0  one_engine_off
```

```
10673  2023-09-09 06:00:00         17.3      97.0  0.0  one_engine_off

[10674 rows x 23 columns]
```

We can see that we have many cases that one single engine is off

```
In [5]: df1.to_csv('/Users/narmina/Downloads/file.csv', index=False)
```

```
In [ ]:
```