In [41]:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
```

IMPORT THE DATASET AND PRE-PROCESSING

Import Train dataset

In [42]:

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your
credentials.
# You might want to remove those credentials before you share the notebook.

if os.environ.get('RUNTIME_ENV_LOCATION_TYPE') == 'external':
    endpoint_80bb5782d084410b9361fad04c8a2f8b = 'https://s3.us.cloud-object-storage.app
domain.cloud'
else:
    endpoint_80bb5782d084410b9361fad04c8a2f8b = 'https://s3.private.us.cloud-object-sto
rage.appdomain.cloud'

client_80bb5782d084410b9361fad04c8a2f8b = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='Vq2UEtWXPE4a_n_ygxHheHFmOQmFPUjDWWAKfV16xNTG',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url=endpoint_80bb5782d084410b9361fad04c8a2f8b)

body = client_80bb5782d084410b9361fad04c8a2f8b.get_object(Bucket='predictivemaintenance
-donotdelete-pr-h8iqsdot8jcrsl',Key='PM_train.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df_train = pd.read_csv(body)
df_train.head()
```

Out[42]:

	0	1	2	3	4	5	6	7	8	9	...	16	17
0	1	1	-0.0007	-0.0004	100	518.67	641.82	1589.70	1400.60	14.62	...	521.66	2388.02
1	1	2	0.0019	-0.0003	100	518.67	642.15	1591.82	1403.14	14.62	...	522.28	2388.07
2	1	3	-0.0043	0.0003	100	518.67	642.35	1587.99	1404.20	14.62	...	522.42	2388.03
3	1	4	0.0007	0.0000	100	518.67	642.35	1582.79	1401.87	14.62	...	522.86	2388.08
4	1	5	-0.0019	-0.0002	100	518.67	642.37	1582.85	1406.22	14.62	...	522.19	2388.04

5 rows × 26 columns

In [43]:

```
# ASSIGN COLUMN NAMES
columns = ['id','cycle','setting1','setting2','setting3','s1','s2','s3','s4','s5','s6',
's7','s8','s9','s10','s11','s12','s13','s14','s15','s16','s17','s18','s19','s20','s21']
df_train.columns = columns
df_train.head()
```

Out[43]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s12
0	1	1	-0.0007	-0.0004	100	518.67	641.82	1589.70	1400.60	14.62	...	521.66
1	1	2	0.0019	-0.0003	100	518.67	642.15	1591.82	1403.14	14.62	...	522.28
2	1	3	-0.0043	0.0003	100	518.67	642.35	1587.99	1404.20	14.62	...	522.42
3	1	4	0.0007	0.0000	100	518.67	642.35	1582.79	1401.87	14.62	...	522.86
4	1	5	-0.0019	-0.0002	100	518.67	642.37	1582.85	1406.22	14.62	...	522.19

5 rows × 26 columns

In [44]:

```
# CHECK FOR NULL VALUES AND REMOVE IF ANY
df_train.isnull().sum()
```

Out[44]:

```
id          0
cycle       0
setting1    0
setting2    0
setting3    0
s1          0
s2          0
s3          0
s4          0
s5          0
s6          0
s7          0
s8          0
s9          0
s10         0
s11         0
s12         0
s13         0
s14         0
s15         0
s16         0
s17         0
s18         0
s19         0
s20         0
s21         0
dtype: int64
```

Import the test dataset.

In [45]:

```
body = client_80bb5782d084410b9361fad04c8a2f8b.get_object(Bucket='predictivemaintenance
-donotdelete-pr-h8iqsdot8jcrsl',Key='PM_test.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df_test = pd.read_csv(body)
df_test.head()
```

Out[45]:

	0	1	2	3	4	5	6	7	8	9	...	16	17
0	1	1	0.0023	0.0003	100	518.67	643.02	1585.29	1398.21	14.62	...	521.72	2388.03
1	1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45	1395.42	14.62	...	522.16	2388.06
2	1	3	0.0003	0.0001	100	518.67	642.46	1586.94	1401.34	14.62	...	521.97	2388.03
3	1	4	0.0042	0.0000	100	518.67	642.44	1584.12	1406.42	14.62	...	521.38	2388.05
4	1	5	0.0014	0.0000	100	518.67	642.51	1587.19	1401.92	14.62	...	522.15	2388.03

5 rows × 26 columns

In [46]:

```
df_test.columns = columns
df_test.head()
```

Out[46]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s12
0	1	1	0.0023	0.0003	100	518.67	643.02	1585.29	1398.21	14.62	...	521.72
1	1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45	1395.42	14.62	...	522.16
2	1	3	0.0003	0.0001	100	518.67	642.46	1586.94	1401.34	14.62	...	521.97
3	1	4	0.0042	0.0000	100	518.67	642.44	1584.12	1406.42	14.62	...	521.38
4	1	5	0.0014	0.0000	100	518.67	642.51	1587.19	1401.92	14.62	...	522.15

5 rows × 26 columns

In [47]:

```
print(df_train.shape)
print(df_test.shape)
```

```
(20631, 26)
(13096, 26)
```

In [48]:

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20631 entries, 0 to 20630
Data columns (total 26 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   id        20631 non-null  int64  
 1   cycle     20631 non-null  int64  
 2   setting1  20631 non-null  float64
 3   setting2  20631 non-null  float64
 4   setting3  20631 non-null  int64  
 5   s1        20631 non-null  float64
 6   s2        20631 non-null  float64
 7   s3        20631 non-null  float64
 8   s4        20631 non-null  float64
 9   s5        20631 non-null  float64
 10  s6        20631 non-null  float64
 11  s7        20631 non-null  float64
 12  s8        20631 non-null  float64
 13  s9        20631 non-null  float64
 14  s10       20631 non-null  float64
 15  s11       20631 non-null  float64
 16  s12       20631 non-null  float64
 17  s13       20631 non-null  float64
 18  s14       20631 non-null  float64
 19  s15       20631 non-null  float64
 20  s16       20631 non-null  float64
 21  s17       20631 non-null  int64  
 22  s18       20631 non-null  int64  
 23  s19       20631 non-null  int64  
 24  s20       20631 non-null  float64
 25  s21       20631 non-null  float64
dtypes: float64(20), int64(6)
memory usage: 4.1 MB
```

In [49]:

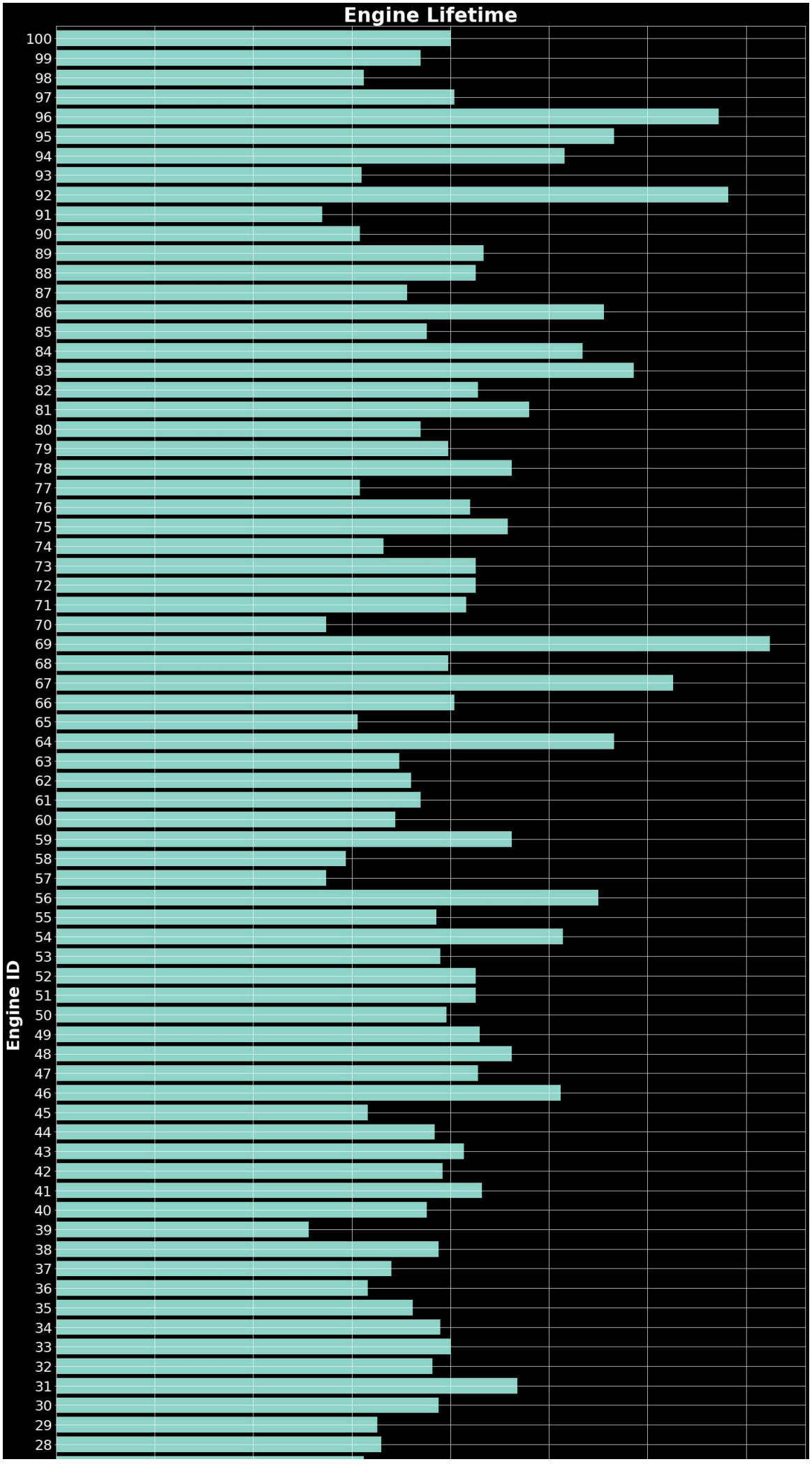
```
df_test.info()
```

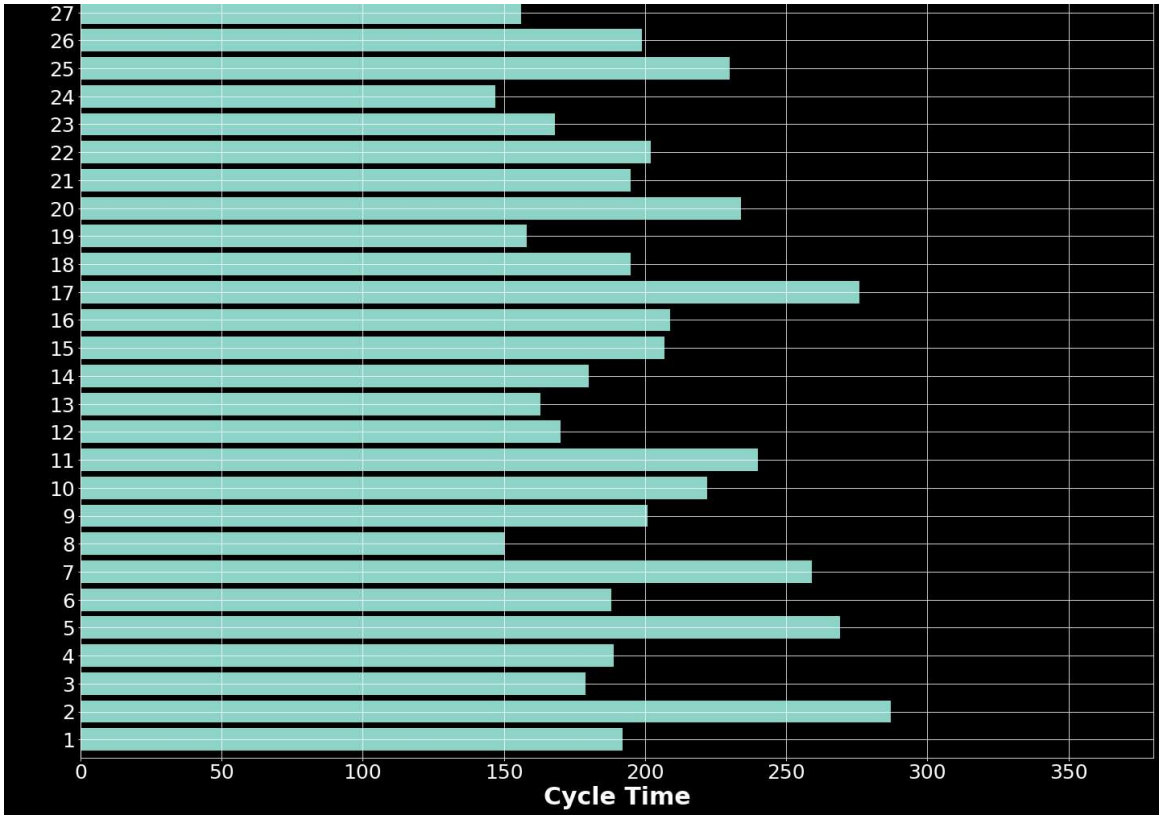
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13096 entries, 0 to 13095
Data columns (total 26 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   id        13096 non-null  int64  
 1   cycle     13096 non-null  int64  
 2   setting1  13096 non-null  float64
 3   setting2  13096 non-null  float64
 4   setting3  13096 non-null  int64  
 5   s1        13096 non-null  float64
 6   s2        13096 non-null  float64
 7   s3        13096 non-null  float64
 8   s4        13096 non-null  float64
 9   s5        13096 non-null  float64
 10  s6        13096 non-null  float64
 11  s7        13096 non-null  float64
 12  s8        13096 non-null  float64
 13  s9        13096 non-null  float64
 14  s10       13096 non-null  float64
 15  s11       13096 non-null  float64
 16  s12       13096 non-null  float64
 17  s13       13096 non-null  float64
 18  s14       13096 non-null  float64
 19  s15       13096 non-null  float64
 20  s16       13096 non-null  float64
 21  s17       13096 non-null  int64  
 22  s18       13096 non-null  int64  
 23  s19       13096 non-null  int64  
 24  s20       13096 non-null  float64
 25  s21       13096 non-null  float64
dtypes: float64(20), int64(6)
memory usage: 2.6 MB
```

ENGINE CYCLE LIFETIME

In [50]:

```
plt.style.use('dark_background')
plt.figure(figsize=(20,50))
ax = df_train.groupby('id')['cycle'].max().plot(kind='barh',width=0.8,stacked = True, a
lign = 'center', rot = 0)
plt.title("Engine Lifetime", fontweight = 'bold', size = 35)
plt.xlabel('Cycle Time', fontweight='bold',size=30)
plt.xticks(size=25)
plt.ylabel('Engine ID',fontweight='bold',size=30)
plt.yticks(size=25)
plt.grid(True)
plt.tight_layout(True)
plt.show()
```


Engine Lifetime
Engine ID
100
99
98
97
96
95
94
93
92
91
90
89
88
87
86
85
84
83
82
81
80
79
78
77
76
75
74
73
72
71
70
69
68
67
66
65
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
48
47
46
45
44
43
42
41
40
39
38
37
36
35
34
33
32
31
30
29
28



PM_truth.txt contains the information about how many more cycles are remaining.

In [51]:

```
body = client_80bb5782d084410b9361fad04c8a2f8b.get_object(Bucket='predictivemaintenance
-donotdelete-pr-h8iqsdot8jcrsl',Key='PM_truth.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

results = pd.read_csv(body)
results.head()
```

Out[51]:

	0
0	112
1	98
2	69
3	82
4	91

In [52]:

```
results['id'] = results.index + 1
```

In [53]:

```
results.head()
```

Out[53]:

	0	id
0	112	1
1	98	2
2	69	3
3	82	4
4	91	5

In [54]:

```
results.columns = ['rul','id']
results.head()
```

Out[54]:

	rul	id
0	112	1
1	98	2
2	69	3
3	82	4
4	91	5

In [55]:

```
rul = pd.DataFrame(df_test.groupby('id')['cycle'].max()).reset_index()
rul.columns = ['id','max']
rul.head()
```

Out[55]:

	id	max
0	1	31
1	2	49
2	3	126
3	4	106
4	5	98

In [56]:

```
results['rtf'] = results['rul'] + rul['max']
results.head()
```

Out[56]:

	rul	id	rtf
0	112	1	143
1	98	2	147
2	69	3	195
3	82	4	188
4	91	5	189

In [57]:

```
results.drop(['rul'],axis = 1, inplace = True)
```

In [58]:

```
results.head()
```

Out[58]:

	id	rtf
0	1	143
1	2	147
2	3	195
3	4	188
4	5	189

In [59]:

```
df_test = df_test.merge(results,on='id',how='left')
df_test
```

Out[59]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...
0	1	1	0.0023	0.0003	100	518.67	643.02	1585.29	1398.21	14.62	...
1	1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45	1395.42	14.62	...
2	1	3	0.0003	0.0001	100	518.67	642.46	1586.94	1401.34	14.62	...
3	1	4	0.0042	0.0000	100	518.67	642.44	1584.12	1406.42	14.62	...
4	1	5	0.0014	0.0000	100	518.67	642.51	1587.19	1401.92	14.62	...
...	...	...	...	...	...	...	...	...	...	...	...
13091	100	194	0.0049	0.0000	100	518.67	643.24	1599.45	1415.79	14.62	...
13092	100	195	-0.0011	-0.0001	100	518.67	643.22	1595.69	1422.05	14.62	...
13093	100	196	-0.0006	-0.0003	100	518.67	643.44	1593.15	1406.82	14.62	...
13094	100	197	-0.0038	0.0001	100	518.67	643.26	1594.99	1419.36	14.62	...
13095	100	198	0.0013	0.0003	100	518.67	642.95	1601.62	1424.99	14.62	...

13096 rows × 27 columns

In [60]:

```
df_test['ttf'] = df_test['rtf']-df_test['cycle']
df_test
```

Out[60]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...
0	1	1	0.0023	0.0003	100	518.67	643.02	1585.29	1398.21	14.62	...
1	1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45	1395.42	14.62	...
2	1	3	0.0003	0.0001	100	518.67	642.46	1586.94	1401.34	14.62	...
3	1	4	0.0042	0.0000	100	518.67	642.44	1584.12	1406.42	14.62	...
4	1	5	0.0014	0.0000	100	518.67	642.51	1587.19	1401.92	14.62	...
...	...	...	...	...	...	...	...	...	...	...	...
13091	100	194	0.0049	0.0000	100	518.67	643.24	1599.45	1415.79	14.62	...
13092	100	195	-0.0011	-0.0001	100	518.67	643.22	1595.69	1422.05	14.62	...
13093	100	196	-0.0006	-0.0003	100	518.67	643.44	1593.15	1406.82	14.62	...
13094	100	197	-0.0038	0.0001	100	518.67	643.26	1594.99	1419.36	14.62	...
13095	100	198	0.0013	0.0003	100	518.67	642.95	1601.62	1424.99	14.62	...

13096 rows × 28 columns

In [61]:

```
df_test.drop(['rtf'],axis=1,inplace=True)
```

In [62]:

```
df_test.head()
```

Out[62]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s1
0	1	1	0.0023	0.0003	100	518.67	643.02	1585.29	1398.21	14.62	...	2388.0
1	1	2	-0.0027	-0.0003	100	518.67	641.71	1588.45	1395.42	14.62	...	2388.0
2	1	3	0.0003	0.0001	100	518.67	642.46	1586.94	1401.34	14.62	...	2388.0
3	1	4	0.0042	0.0000	100	518.67	642.44	1584.12	1406.42	14.62	...	2388.0
4	1	5	0.0014	0.0000	100	518.67	642.51	1587.19	1401.92	14.62	...	2388.0

5 rows × 27 columns

In [63]:

```
df_train['ttf'] = df_train.groupby(['id'])['cycle'].transform(max)-df_train['cycle']
df_train.head()
```

Out[63]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s1
0	1	1	-0.0007	-0.0004	100	518.67	641.82	1589.70	1400.60	14.62	...	2388.0
1	1	2	0.0019	-0.0003	100	518.67	642.15	1591.82	1403.14	14.62	...	2388.0
2	1	3	-0.0043	0.0003	100	518.67	642.35	1587.99	1404.20	14.62	...	2388.0
3	1	4	0.0007	0.0000	100	518.67	642.35	1582.79	1401.87	14.62	...	2388.0
4	1	5	-0.0019	-0.0002	100	518.67	642.37	1582.85	1406.22	14.62	...	2388.0

5 rows × 27 columns

In [64]:

```
print(df_train.columns)
print(df_test.columns)
```

```
Index(['id', 'cycle', 'setting1', 'setting2', 'setting3', 's1', 's2', 's
3',
       's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's1
4',
       's15', 's16', 's17', 's18', 's19', 's20', 's21', 'ttf'],
      dtype='object')
Index(['id', 'cycle', 'setting1', 'setting2', 'setting3', 's1', 's2', 's
3',
       's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's1
4',
       's15', 's16', 's17', 's18', 's19', 's20', 's21', 'ttf'],
      dtype='object')
```

In [65]:

```
df_train['ttf'].value_counts()
```

Out[65]:

```
0      100
123    100
121    100
89     100
73     100
      ...
341      1
356      1
355      1
354      1
351      1
Name: ttf, Length: 362, dtype: int64
```

In [66]:

```
train = df_train.copy()
test = df_test.copy()
period = 30
df_train['label_bc'] = df_train['ttf'].apply(lambda x: 1 if x <= period else 0)
df_test['label_bc'] = df_test['ttf'].apply(lambda x: 1 if x <= period else 0)
df_train.head()
```

Out[66]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s1
0	1	1	-0.0007	-0.0004	100	518.67	641.82	1589.70	1400.60	14.62	...	8138.6
1	1	2	0.0019	-0.0003	100	518.67	642.15	1591.82	1403.14	14.62	...	8131.4
2	1	3	-0.0043	0.0003	100	518.67	642.35	1587.99	1404.20	14.62	...	8133.2
3	1	4	0.0007	0.0000	100	518.67	642.35	1582.79	1401.87	14.62	...	8133.8
4	1	5	-0.0019	-0.0002	100	518.67	642.37	1582.85	1406.22	14.62	...	8133.8

5 rows × 28 columns

In [67]:

```
df_train['label_bc'].value_counts()
```

Out[67]:

```
0    17531
1     3100
Name: label_bc, dtype: int64
```

In [68]:

```
df_test['label_bc'].value_counts()
```

Out[68]:

```
0    12764
1      332
Name: label_bc, dtype: int64
```

In [69]:

```
features_col_name=['setting1', 'setting2', 'setting3', 's1', 's2', 's3', 's4', 's5', 's
6', 's7', 's8', 's9', 's10', 's11',
                   's12', 's13', 's14', 's15', 's16', 's17', 's18', 's19', 's20', 's21'
]
target_col_name='label_bc'
```

In [70]:

```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
df_train[features_col_name] = sc.fit_transform(df_train[features_col_name])
df_test[features_col_name] = sc.fit_transform(df_test[features_col_name])
```

In [71]:

```
df_train.head()
```

Out[71]:

	id	cycle	setting1	setting2	setting3	s1	s2	s3	s4	s5	...	s
0	1	1	0.459770	0.166667	0.0	0.0	0.183735	0.406802	0.309757	0.0	...	0.1996
1	1	2	0.609195	0.250000	0.0	0.0	0.283133	0.453019	0.352633	0.0	...	0.1628
2	1	3	0.252874	0.750000	0.0	0.0	0.343373	0.369523	0.370527	0.0	...	0.1717
3	1	4	0.540230	0.500000	0.0	0.0	0.343373	0.256159	0.331195	0.0	...	0.1748
4	1	5	0.390805	0.333333	0.0	0.0	0.349398	0.257467	0.404625	0.0	...	0.1747

5 rows × 28 columns

In [72]:

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20631 entries, 0 to 20630
Data columns (total 28 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   id        20631 non-null  int64  
 1   cycle     20631 non-null  int64  
 2   setting1  20631 non-null  float64
 3   setting2  20631 non-null  float64
 4   setting3  20631 non-null  float64
 5   s1        20631 non-null  float64
 6   s2        20631 non-null  float64
 7   s3        20631 non-null  float64
 8   s4        20631 non-null  float64
 9   s5        20631 non-null  float64
 10  s6        20631 non-null  float64
 11  s7        20631 non-null  float64
 12  s8        20631 non-null  float64
 13  s9        20631 non-null  float64
 14  s10       20631 non-null  float64
 15  s11       20631 non-null  float64
 16  s12       20631 non-null  float64
 17  s13       20631 non-null  float64
 18  s14       20631 non-null  float64
 19  s15       20631 non-null  float64
 20  s16       20631 non-null  float64
 21  s17       20631 non-null  float64
 22  s18       20631 non-null  float64
 23  s19       20631 non-null  float64
 24  s20       20631 non-null  float64
 25  s21       20631 non-null  float64
 26  ttf       20631 non-null  int64  
 27  label_bc  20631 non-null  int64  
dtypes: float64(24), int64(4)
memory usage: 4.4 MB
```

import seaborn as sns sns.pairplot(df_train)

In [73]:

```
df_train.iloc[0,:]
```

Out[73]:

```
id            1.000000
cycle         1.000000
setting1      0.459770
setting2      0.166667
setting3      0.000000
s1            0.000000
s2            0.183735
s3            0.406802
s4            0.309757
s5            0.000000
s6            1.000000
s7            0.726248
s8            0.242424
s9            0.109755
s10           0.000000
s11           0.369048
s12           0.633262
s13           0.205882
s14           0.199608
s15           0.363986
s16           0.000000
s17           0.333333
s18           0.000000
s19           0.000000
s20           0.713178
s21           0.724662
ttf         191.000000
label_bc      0.000000
Name: 0, dtype: float64
```

In [74]:

```
x_train = df_train.iloc[:,:-1].values
y_train = df_train.iloc[:,-1].values
```

In [75]:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(solver='liblinear')
model.fit(x_train,y_train)
```

Out[75]:

```
LogisticRegression(solver='liblinear')
```

In [78]:

```
x_train[0]
```

Out[78]:

```
array([1.00000000e+00, 1.00000000e+00, 4.59770115e-01, 1.66666667e-01,
       0.00000000e+00, 0.00000000e+00, 1.83734940e-01, 4.06801831e-01,
       3.09756921e-01, 0.00000000e+00, 1.00000000e+00, 7.26247987e-01,
       2.42424242e-01, 1.09755003e-01, 0.00000000e+00, 3.69047619e-01,
       6.33262260e-01, 2.05882353e-01, 1.99607803e-01, 3.63986149e-01,
       0.00000000e+00, 3.33333333e-01, 0.00000000e+00, 0.00000000e+00,
       7.13178295e-01, 7.24661696e-01, 1.91000000e+02])
```

In [79]:

```
!pip install ibm_watson_machine_learning
```

```
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorag
e/dhcrypto.py:16: CryptographyDeprecationWarning: int_from_bytes is deprec
ated, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
/opt/conda/envs/Python-3.7-OpenCE/lib/python3.7/site-packages/secretstorag
e/util.py:25: CryptographyDeprecationWarning: int_from_bytes is deprecate
d, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/e
nvs/Python-3.7-OpenCE/lib/python3.7/site-packages (1.0.105)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.7-Open
CE/lib/python3.7/site-packages (from ibm_watson_machine_learning) (2.22.0)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.7-Ope
nCE/lib/python3.7/site-packages (from ibm_watson_machine_learning) (20.4)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.7-Open
CE/lib/python3.7/site-packages (from ibm_watson_machine_learning) (0.8.3)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ibm_watson_machine_learning) (2021.5.3
0)
Requirement already satisfied: pandas<1.3.0,>=0.24.2 in /opt/conda/envs/Py
thon-3.7-OpenCE/lib/python3.7/site-packages (from ibm_watson_machine_learn
ing) (1.0.5)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.7-OpenC
E/lib/python3.7/site-packages (from ibm_watson_machine_learning) (1.25.9)
Requirement already satisfied: ibm-cos-sdk==2.7.* in /opt/conda/envs/Pytho
n-3.7-OpenCE/lib/python3.7/site-packages (from ibm_watson_machine_learnin
g) (2.7.0)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /opt/conda/envs/Py
thon-3.7-OpenCE/lib/python3.7/site-packages (from requests->ibm_watson_mac
hine_learning) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in /opt/conda/envs/Python-3.
7-OpenCE/lib/python3.7/site-packages (from requests->ibm_watson_machine_le
arning) (2.8)
Requirement already satisfied: pyparsing>=2.0.2 in /opt/conda/envs/Python-
3.7-OpenCE/lib/python3.7/site-packages (from packaging->ibm_watson_machine
_learning) (2.4.7)
Requirement already satisfied: six in /opt/conda/envs/Python-3.7-OpenCE/li
b/python3.7/site-packages (from packaging->ibm_watson_machine_learning)
(1.15.0)
Requirement already satisfied: python-dateutil>=2.6.1 in /opt/conda/envs/P
ython-3.7-OpenCE/lib/python3.7/site-packages (from pandas<1.3.0,>=0.24.2->
ibm_watson_machine_learning) (2.8.1)
Requirement already satisfied: numpy>=1.13.3 in /opt/conda/envs/Python-3.7
-OpenCE/lib/python3.7/site-packages (from pandas<1.3.0,>=0.24.2->ibm_watso
n_machine_learning) (1.19.2)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/envs/Python-3.7-
OpenCE/lib/python3.7/site-packages (from pandas<1.3.0,>=0.24.2->ibm_watson
_machine_learning) (2020.1)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/P
ython-3.7-OpenCE/lib/python3.7/site-packages (from ibm-cos-sdk==2.7.*->ibm
_watson_machine_learning) (0.9.4)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.7.0 in /opt/cond
a/envs/Python-3.7-OpenCE/lib/python3.7/site-packages (from ibm-cos-sdk==2.
7.*->ibm_watson_machine_learning) (2.7.0)
Requirement already satisfied: ibm-cos-sdk-core==2.7.0 in /opt/conda/envs/
Python-3.7-OpenCE/lib/python3.7/site-packages (from ibm-cos-sdk==2.7.*->ib
m_watson_machine_learning) (2.7.0)
Requirement already satisfied: docutils<0.16,>=0.10 in /opt/conda/envs/Pyt
```

```
hon-3.7-OpenCE/lib/python3.7/site-packages (from ibm-cos-sdk-core==2.7.0->
ibm-cos-sdk==2.7.*->ibm_watson_machine_learning) (0.15.2)
```

In [80]:

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey":"_2Cr1-7P6-ZnBRqLRNhlvmEo5XP6GBG1-_qkCuLTXDvM"
}
client = APIClient(wml_credentials)
```

In [81]:

```
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']["name"] == space_
name)['metadata']['id'])
```

In [82]:

```
space_uid = guid_from_space_name(client, 'models')
print("Space UID = " + space_uid)
```

```
Space UID = 80da3fe5-581f-4d4b-9014-6ad2c4abbac0
```

In [83]:

```
client.set.default_space(space_uid)
```

Out[83]:

```
'SUCCESS'
```

In [84]:

```
client.software_specifications.list()
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49  base
autoai-ts_3.8-py3.8             2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6           2b73a275-7cbf-420b-a912-eae7f436e0bc  base
pytorch_1.2-py3.6               2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                 2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt      32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37            36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                 390d21f8-e58b-4fac-9c55-d7ceda621326  base
xgboost_0.82-py3.6              39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt      40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
autoai-obm_3.0                  42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
spark-mllib_2.4-r_3.6           49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6              4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6          50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
spark-mllib_2.4-scala_2.11      55a70f99-7320-4be5-9fb9-9edb5a443af5  base
autoai-obm_2.0                  5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1               5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
autoai-kb_3.1-py3.7             632d4b22-10aa-5180-88f0-f52dfb6444d7  base
pytorch-onnx_1.7-py3.8          634d3cdc-b562-5bf9-a2d4-ea90a478456b  base
spark-mllib_2.3-r_3.6           6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c  base
tensorflow_2.4-py3.7            65e171d7-72d1-55d9-8ebb-f813d620c9bb  base
spss-modeler_18.2               687eddc9-028a-4117-b9dd-e57b36f1efa5  base
pytorch-onnx_1.2-py3.6          692a6a4d-2c4d-45ff-a1ed-b167ee55469a  base
do_12.9                         75a3a4b0-6aa0-41b3-a618-48b1f56332a6  base
spark-mllib_2.3-scala_2.11      7963efe5-bbec-417e-92cf-0574e21b4e8d  base
spark-mllib_2.4-py37            7abc992b-b685-532b-a122-a396a3cdbaab  base
caffe_1.0-py3.6                 7bb3dbe2-da6e-4145-918d-b6d84aa93b6b  base
pytorch-onnx_1.7-py3.7          812c6631-42b7-5613-982b-02098e6c909c  base
cuda-py3.6                      82c79ece-4d12-40e6-8787-a7b9e0f62770  base
tensorflow_1.15-py3.6-horovod   8964680e-d5e4-5bb8-919b-8342c6c0dfd8  base
hybrid_0.1                      8c1a58c6-62b5-4dc4-987a-df751c2756b6  base
pytorch-onnx_1.3-py3.7          8d5d8a87-a912-54cf-81ec-3914adaa988d  base
caffe-ibm_1.0-py3.6             8d863266-7927-4d1e-97d7-56a7f4c0a19b  base
spss-modeler_17.1               902d0051-84bd-4af6-ab6b-8f6aa6fdeabb  base
do_12.10                        9100fd72-8159-4eb9-8a0b-a87e12eefa36  base
do_py3.7                        9447fa8b-2051-4d24-9eef-5acb0e3c59f8  base
spark-mllib_3.0-r_3.6           94bb6052-c837-589d-83f1-f4142f219e32  base
------------------------------  ------------------------------------  ----
Note: Only first 50 records were displayed. To display more use 'limit' pa
rameter.
```

In [90]:

```
software_spec_uid = client.software_specifications.get_uid_by_name("default_py3.7_opence")
software_spec_uid
```

Out[90]:

```
'c2057dd4-f42c-5f77-a02f-72bdbd3282c9'
```

In [91]:

```
model_details = client.repository.store_model(model=model,meta_props={
    client.repository.ModelMetaNames.NAME : "Engine_Model",
    client.repository.ModelMetaNames.TYPE : "scikit-learn_0.23",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID : software_spec_uid
})
model_id = client.repository.get_model_uid(model_details)
```

In [92]:

```
model_id
```

Out[92]:

```
'7ffd21b5-f98e-4168-b7d4-cc7703fc4cf6'
```

In [ ]:

```

```

In [88]:

```
x_train[0]
```

Out[88]:

```
array([1.00000000e+00, 1.00000000e+00, 4.59770115e-01, 1.66666667e-01,
       0.00000000e+00, 0.00000000e+00, 1.83734940e-01, 4.06801831e-01,
       3.09756921e-01, 0.00000000e+00, 1.00000000e+00, 7.26247987e-01,
       2.42424242e-01, 1.09755003e-01, 0.00000000e+00, 3.69047619e-01,
       6.33262260e-01, 2.05882353e-01, 1.99607803e-01, 3.63986149e-01,
       0.00000000e+00, 3.33333333e-01, 0.00000000e+00, 0.00000000e+00,
       7.13178295e-01, 7.24661696e-01, 1.91000000e+02])
```

In [89]:

```
model.predict([[1.00000000e+00, 1.00000000e+00, 4.59770115e-01, 1.66666667e-01,
       0.00000000e+00, 0.00000000e+00, 1.83734940e-01, 4.06801831e-01,
       3.09756921e-01, 0.00000000e+00, 1.00000000e+00, 7.26247987e-01,
       2.42424242e-01, 1.09755003e-01, 0.00000000e+00, 3.69047619e-01,
       6.33262260e-01, 2.05882353e-01, 1.99607803e-01, 3.63986149e-01,
       0.00000000e+00, 3.33333333e-01, 0.00000000e+00, 0.00000000e+00,
       7.13178295e-01, 7.24661696e-01, 1.91000000e+02]])
```

Out[89]:

```
array([0], dtype=int64)
```

In [ ]:

IGNORE CODE BELOW

import joblib joblib.dump(model,'engine_model.sav')

x_test = df_test.iloc[:,:-1].values y_test = df_test.iloc[:,-1:].values

y_predlog = model.predict(x_test)

from sklearn.metrics import accuracy_score accuracy_score(y_predlog,y_test)

TESTING OUR MODEL - User input Case

joblib.dump(sc,'MinMax.sav')

minmaxTransform = joblib.load('MinMax.sav') load_model = joblib.load('engine_model.sav')

sample = minmaxTransform.transform([[0.0011, -0.0001, 100.0, 518.67, 642.61, 1587.78, 1400.70, 14.62, 21.61, 554.31, 2388.05, 9041.12, 1.30, 47.46, 522.28, 2388.05, 8128.59, 8.4099, 0.03, 392, 2388, 100.00, 39.00, 23.3325]])

sample

eid = 1 cyc = 20 timetf = 123 example = [[eid,cyc,sample[0][0], sample[0][1], sample[0][2], sample[0][3], sample[0][4], sample[0][5], sample[0][6], sample[0][7], sample[0][8], sample[0][9], sample[0][10], sample[0][11], sample[0][12], sample[0][13], sample[0][14], sample[0][15], sample[0][16], sample[0][17], sample[0][18], sample[0][19], sample[0][20], sample[0][21], sample[0][22], sample[0][23], timetf]] x = load_model.predict(example) print(x)

In [ ]: