outputRate

March 29, 2020

1 Determining the required output rate for datasets in Group A

Set 2 contains 2 datasets of data collected of 2 identical cells with the same load profile. Reflective of rail operations.

1.0.1 Import necessary libraries

[1]:

```
import numpy as np
import pandas as pd
import copy
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# import codebase
import thermalModel_main as tmm
import thermalModel_groupB as tm_gb

import importlib
importlib.reload(tmm)
importlib.reload(tm_gb)
```

```
Using TensorFlow backend.
```

[1]:

```
<module 'thermalModel_groupB' from
'C:\\Users\\user\\Anaconda3\\lib\\thermalModel_groupB.py'>
```

[2]:

```
AhData_2097_df = tm_gb.load_csv(filename = 'LDPRF_2097.csv',
                                features_list =␣
 ↪['second','AhCha','AhDch','Current','Voltage','Amb','Temp'], mode = 2)

AhData_2098_df = tm_gb.load_csv(filename = 'LDPRF_2098.csv',
                                features_list =␣
 ↪['second','AhCha','AhDch','Current','Voltage','Amb','Temp'], mode = 2)
```

```
C:\Users\user\Anaconda3\lib\thermalModel_groupB.py:47: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  df['second'][set_index[index]:set_index[index+1]] =
df['second'][set_index[index]:set_index[index+1]] + second_increment[index]
C:\Users\user\Anaconda3\lib\thermalModel_groupB.py:49: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  df['second'][set_index[index]:] = df['second'][set_index[index]:] +
second_increment[index]
C:\Users\user\Anaconda3\lib\thermalModel_groupB.py:56: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  df['second'][set_index[index]:] = df['second'][set_index[index]:] +
seconds_summation[index]
```

[3]:

```python
print("Shape: {}".format(AhData_2097_df.shape))
print("Shape: {}".format(AhData_2098_df.shape))
```

```
Shape: (435839, 7)
Shape: (435839, 7)
```

[4]:

```python
AhData_2097_df.describe()
```

[4]:

```
              second          AhCha          AhDch        Current  \
count  435839.000000  435839.000000  435839.000000  435839.000000
mean    24468.740483      -0.595961       3.775370     126.363856
std     11440.765250      85.854861       0.091213      72.924632
min         0.000000    -177.639340       3.536830       0.000000
25%     14564.850000       0.009580       3.730960      64.452000
50%     24470.300000       0.009580       3.766810     126.039000
75%     34375.750000       0.019150       3.807290     187.997000
max     44280.800000     223.268950       4.160100     252.040000

             Voltage          Amb           Temp
count  435839.000000  4.358390e+05  435839.000000
mean      144.644944  2.579465e+01      34.312581
std        74.703530  2.402277e-10       2.060416
min         0.000000  2.579465e+01      25.794650
25%        81.299000  2.579465e+01      33.008410
50%       145.061000  2.579465e+01      35.085100
75%       208.479000  2.579465e+01      35.850190
max       272.253000  2.579465e+01      36.724590
```

[5]:
```
AhData_2098_df.describe()
```

[5]:
```
              second          AhCha          AhDch        Current  \
count  435839.000000  435839.000000  435839.000000  435839.000000
mean    20773.140869      -0.497548       3.782469     126.437695
std     11367.928295      85.732075       0.086605      72.927347
min         0.000000    -176.603480       3.557440       0.000000
25%     10905.200000       0.009560       3.741410      64.531000
50%     20810.600000       0.009560       3.773560     126.152000
75%     30596.250000       0.009560       3.813390     188.089000
max     40501.400000     222.893370       4.161120     252.044000

             Voltage           Amb           Temp
count  435839.000000  4.358390e+05  435839.000000
mean      143.968215  2.626750e+01      34.934164
std        74.268447  6.600594e-11       1.938317
min         0.000000  2.626750e+01      26.267500
25%        80.996500  2.626750e+01      33.803260
50%       144.421000  2.626750e+01      35.741030
75%       207.443000  2.626750e+01      36.386950
max       270.765000  2.626750e+01      37.032870
```

1.0.2 Check original rates

[6]:
```
df = copy.deepcopy(AhData_2097_df)

df['temp_difference'] = df['Temp'].diff(periods=1)
df['temp_difference'] = df['temp_difference'].abs()

row_indices=df[(df['temp_difference'] == 0.0)].index
df.drop(row_indices, inplace=True)
df.dropna(axis=0, inplace=True)
df.reset_index(drop=True, inplace=True)

df.describe()
print("Shape: {}".format(df.shape))
```

```
Shape: (2877, 8)
```

[7]:
```
df1 = copy.deepcopy(AhData_2098_df)

df1['temp_difference'] = df1['Temp'].diff(periods=1)
df1['temp_difference'] = df1['temp_difference'].abs()

row_indices=df1[(df1['temp_difference'] == 0.0)].index
df1.drop(row_indices, inplace=True)
df1.dropna(axis=0, inplace=True)
```

```python
df1.reset_index(drop=True, inplace=True)

df1.describe()
print("Shape: {}".format(df1.shape))
```

```
Shape: (3776, 8)
```

[8]:

```python
df.head(20)
```

[8]:

```
    second     AhCha    AhDch  Current  Voltage       Amb      Temp  \
0     80.0 -46.00462  4.04773      0.0    1.021  25.79465  25.90395
1    100.0 -45.99504  4.03988      0.0    1.278  25.79465  25.79465
2    110.0 -46.00462  4.03605      0.0    1.405  25.79465  25.90395
3    140.0 -46.00462  4.02498      0.0    1.788  25.79465  26.01325
4    230.0 -45.99504  3.99437      0.0    2.938  25.79465  26.12255
5    250.0 -45.99504  3.98792      0.0    3.194  25.79465  26.01325
6    280.0 -46.00462  3.97826      0.0    3.578  25.79465  26.12255
7    300.0 -45.99504  3.97181      0.0    3.833  25.79465  26.23185
8    330.0 -46.00462  3.96235      0.0    4.216  25.79465  26.34115
9    350.0 -46.00462  3.95611      0.0    4.471  25.79465  26.23185
10   360.0 -46.00462  3.95288      0.0    4.599  25.79465  26.34115
11   400.0 -45.99504  3.94080      0.0    5.110  25.79465  26.45045
12   410.0 -46.00462  3.93778      0.0    5.239  25.79465  26.34115
13   420.0 -46.00462  3.93476      0.0    5.366  25.79465  26.45045
14   440.0 -46.00462  3.92872      0.0    5.621  25.79465  26.34115
15   450.0 -46.00462  3.92590      0.0    5.749  25.79465  26.55975
16   460.0 -46.00462  3.92268      0.0    5.877  25.79465  26.45045
17   470.0 -45.99504  3.91986      0.0    6.005  25.79465  26.55975
18   540.0 -45.99504  3.89952      0.0    6.899  25.79465  26.66904
19   550.0 -46.00462  3.89670      0.0    7.027  25.79465  26.77834

    temp_difference
0           0.10930
1           0.10930
2           0.10930
3           0.10930
4           0.10930
5           0.10930
6           0.10930
7           0.10930
8           0.10930
9           0.10930
10          0.10930
11          0.10930
12          0.10930
13          0.10930
14          0.10930
15          0.21860
```

```
16          0.10930
17          0.10930
18          0.10929
19          0.10930
```

[9]:
```
df1.head(20)
```

[9]:
```
    second     AhCha    AhDch  Current  Voltage      Amb      Temp  \
0     10.0 -45.99527  4.09299      0.0    0.128  26.2675  26.37516
1     50.0 -45.99527  4.06429      0.0    0.639  26.2675  26.26750
2     70.0 -45.99527  4.05519      0.0    0.894  26.2675  26.37516
3    140.0 -45.99527  4.02830      0.0    1.789  26.2675  26.48281
4    170.0 -45.99527  4.01779      0.0    2.172  26.2675  26.59046
5    220.0 -45.99527  4.00100      0.0    2.811  26.2675  26.69812
6    270.0 -45.98571  3.98463      0.0    3.449  26.2675  26.80577
7    280.0 -45.99527  3.98139      0.0    3.577  26.2675  26.91343
8    300.0 -45.99527  3.97492      0.0    3.833  26.2675  27.02108
9    370.0 -45.99527  3.95309      0.0    4.727  26.2675  27.12873
10   460.0 -45.99527  3.92580      0.0    5.877  26.2675  27.23639
11   520.0 -45.98571  3.90841      0.0    6.643  26.2675  27.34404
12   590.0 -45.99527  3.88839      0.0    7.537  26.2675  27.45169
13   620.0 -45.99527  3.88011      0.0    7.921  26.2675  27.55935
14   640.0 -45.99527  3.87465      0.0    8.176  26.2675  27.66700
15   730.0 -45.99527  3.85039      0.0    9.326  26.2675  27.77465
16   740.0 -45.99527  3.84756      0.0    9.454  26.2675  27.66700
17   760.0 -45.98571  3.84250      0.0    9.709  26.2675  27.77465
18   810.0 -45.98571  3.82956      0.0   10.348  26.2675  27.88231
19   820.0 -45.99527  3.82673      0.0   10.476  26.2675  27.77465

    temp_difference
0           0.10766
1           0.10766
2           0.10766
3           0.10765
4           0.10765
5           0.10766
6           0.10765
7           0.10766
8           0.10765
9           0.10765
10          0.10766
11          0.10765
12          0.10765
13          0.10766
14          0.10765
15          0.10765
16          0.10765
17          0.10765
```

```
18          0.10766
19          0.10766
```

[10]:

```
def cumsum_breach(x, target):
    total = 0
    for i, y in enumerate(x):
        total += y
        if total >= target:
            yield i
            total = 0

# list_for_cumsum = df['temp_difference'].values.tolist()
list_for_cumsum1 = df['temp_difference'].to_numpy(dtype=None, copy=True)
list_1 = list(np.around(list_for_cumsum1,2))
list_toKeep1 = list(cumsum_breach(list_1, 0.3)) # change this to change the␣
 ↪magnitude of cummulative change
list_2 = [x for x in range(0, len(df))]
list_toDrop1 = [x for x in list_2 if x not in list_toKeep1]
print("Number of elements to keep: {}".format(len(list_toKeep1)))
print("Number of elements to drop: {}".format(len(list_toDrop1)))
df_reduced = df.drop(list_toDrop1)
df_reduced = df_reduced.drop(columns = ['temp_difference'])
df_reduced.describe()
df_reduced.to_csv('groupB_reduced_dataset.csv')

# list_for_cumsum = df['temp_difference'].values.tolist()
list_for_cumsum3 = df1['temp_difference'].to_numpy(dtype=None, copy=True)
list_3 = list(np.around(list_for_cumsum3,2))
list_toKeep3 = list(cumsum_breach(list_3, 0.3)) # change this to change the␣
 ↪magnitude of cummulative change
list_4 = [x for x in range(0, len(df1))]
list_toDrop3 = [x for x in list_4 if x not in list_toKeep3]
print("Number of elements to keep: {}".format(len(list_toKeep3)))
print("Number of elements to drop: {}".format(len(list_toDrop3)))
df_reduced1 = df1.drop(list_toDrop3)
df_reduced1 = df_reduced1.drop(columns = ['temp_difference'])
df_reduced1.describe()
df_reduced1.to_csv('groupB1_reduced_dataset.csv')
```

```
Number of elements to keep: 961
Number of elements to drop: 1916
Number of elements to keep: 1259
Number of elements to drop: 2517
```

[11]:

```
df_reduced.head(5)
```

[11]:

```
    second     AhCha    AhDch  Current  Voltage       Amb      Temp
2    110.0 -46.00462  4.03605      0.0    1.405  25.79465  25.90395
```

```
5    250.0 -45.99504  3.98792      0.0    3.194  25.79465  26.01325
8    330.0 -46.00462  3.96235      0.0    4.216  25.79465  26.34115
11   400.0 -45.99504  3.94080      0.0    5.110  25.79465  26.45045
14   440.0 -46.00462  3.92872      0.0    5.621  25.79465  26.34115
```

[12]:

```
df_reduced1.head(5)
```

[12]:

```
    second      AhCha    AhDch  Current  Voltage      Amb       Temp
2     70.0 -45.99527  4.05519      0.0    0.894  26.2675  26.37516
5    220.0 -45.99527  4.00100      0.0    2.811  26.2675  26.69812
8    300.0 -45.99527  3.97492      0.0    3.833  26.2675  27.02108
11   520.0 -45.98571  3.90841      0.0    6.643  26.2675  27.34404
14   640.0 -45.99527  3.87465      0.0    8.176  26.2675  27.66700
```

[13]:

```
print('Temperature changes with an cummulative magnitude of 0.3 degrees every:')
df_reduced['second'].diff(periods=1).describe()
```

```
Temperature changes with an cummulative magnitude of 0.3 degrees every:
```

[13]:

```
count    960.000000
mean      45.946250
std       35.863897
min        2.100000
25%       22.775000
50%       37.200000
75%       58.825000
max      400.000000
Name: second, dtype: float64
```

[14]:

```
print('Temperature changes with an cummulative magnitude of 0.3 degrees every:')
df_reduced1['second'].diff(periods=1).abs().describe()
```

```
Temperature changes with an cummulative magnitude of 0.3 degrees every:
```

[14]:

```
count    1258.000000
mean       37.297774
std        95.776393
min         0.600000
25%        13.000000
50%        25.000000
75%        45.875000
max      3180.000000
Name: second, dtype: float64
```