Meeting #4

09/30/21

Omar Luna

Deliverables

- Epoch detection function
- Apply changePeaks_epochDetect() to data
- Plot biometric data over time
- Overlay epoch borders
- (Optional) Epoch_detection method

Methodology and Learnings

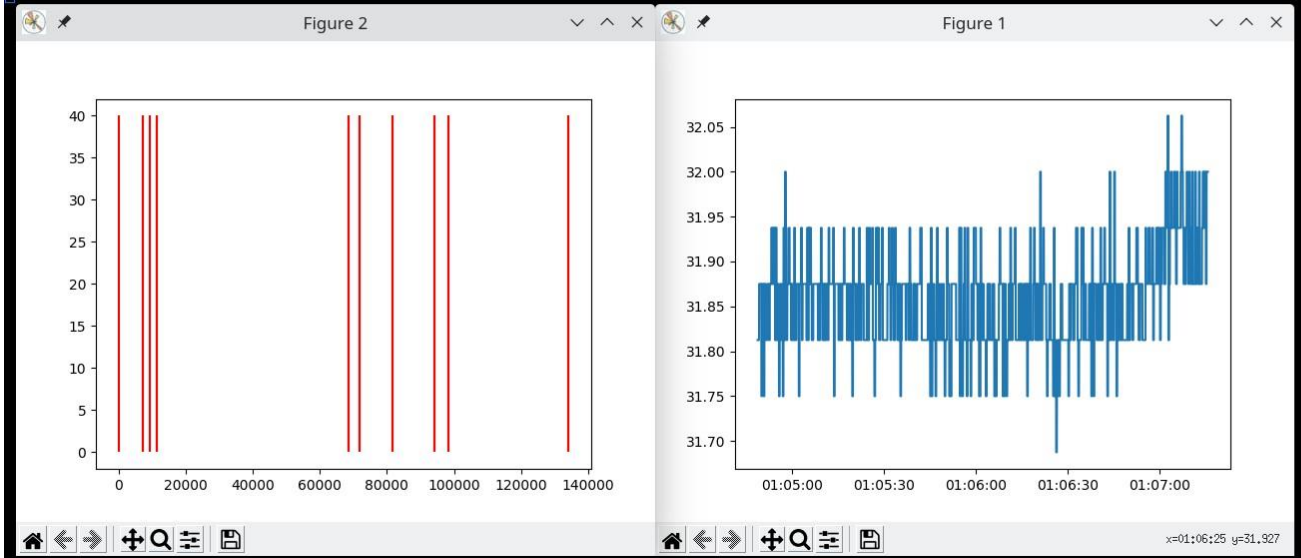
- Epoch detection
  - used previous deliverable to get data
    - Plotted data vs only "time" data I could implement
  - Used changePeaks_epochDetec
    - Retrieved 9 keys
    - Tuples of data marking assumed boundaries
    - Used the tuples as points to plot epoch borders

Learning

- Python dictionaries
- Anaconda

Results

```
(csproject) [omar@olA15 functions]$ python display_epoch.py python
slice(2999, -2999, None)
dict_values([[0, 11556], [11556, 98203], [98203, 94136], [94136, 7253], [7253, 134177], [134177, 81678], [81678, 71708], [71708, 68717], [68717, 9438]])
```



display_epoch.py | changePeaks_epochDetection.py | display_data.py

```
import changePeaks_epochDetection as epochdetect
import pandas as pds
import numpy as npy
from matplotlib import pyplot as plt

# Read in data
Y = dd.returnData('temp')
# Convert to 1D numpy array
Y.to_numpy()
# Plot "time" vs Temp
fig1 = plt.figure(1)
time = [day_t.datetime.now() + day_t.timedelta(milliseconds=i) for i in range(146884)]
plt.plot(time, Y)

# Use epoch function to return a dictionary
epoch_dict = epochdetect.changePeaks_epochDetect(Y, 10, 1, 2000)
print(epoch_dict.values())
myL = epoch_dict.values()
# Sort dictionary values and create tuples for range
myL = sorted(myL)
x, y = zip(*myL)

xepoch_points = []
for i in range(len(myL)):
    xepoch_points.append(myL[i][0])
    xepoch_points.append(myL[i][1])
# Plot vertical lines for epoch_division
fig2 = plt.figure(2)


for i in range(len(xepoch_points)):
    plt.vlines(x = xepoch_points[i], ymin =0, ymax = 40, colors='red', label = 'epoch_point')
plt.show()

```