****

* Description *

****

Write a sensor data analytics program to build the relationship model between the sensor data and the parameters (S, D), and predict S and D from future sensor data. Python language is expected for the sensor data analytics program. The goal is to achieve a low MAE (less than 3 ideally) for two parameters (S, D) on the test data set and show a matching trend comparison plot of labels and prediction results (see main_plot.py as the example]).

In the data file, each row includes sensor data (10 seconds * 100Hz) + ID + Time + H + R + S + D, which may be visualized with the command "./view_data.py xxx.npy 6". H, R, S and D are heartrate, respiratory rate, systolic and diastolic blood pressure. Here 10 seconds * 100Hz means that each row contains 10 seconds data and the data sampling rate is 100Hz (e.g., 100 data points per second).

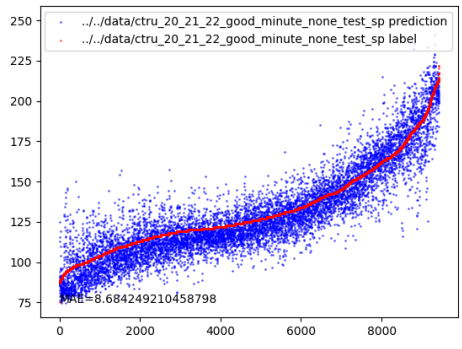
The data analytics program may be based on advanced signal processing, unsupervised /supervised machine learning or deep learning, or their combinations. If you choose to perform the supervised training, please use the train data set (*_train.npy) for training, and do not include the test data set (*_test.npy) in training, which shall be used for test only.

****

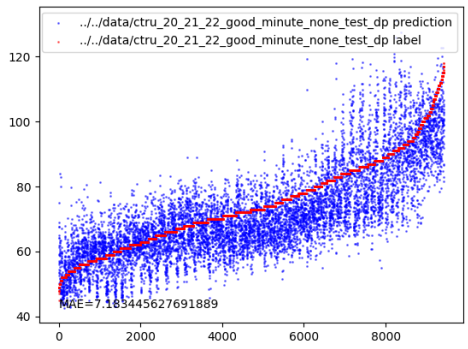
* Delivery *

****

Please save your trained model in a file and write a prediction program that reads the saved model, takes the test data set as the input, outputs the prediction result (S, D), and prints their MAE and use plot_2vectors() function to save the plots of the label and prediction result (see example plots below and example code in main_plot.py), so that we may verify your result by calling your prediction function only, without having to run your training program.



S: prediction vs label



D: prediction vs label

2. The goal is to achieve MAE <=3 for the parameters (S, D) on the test data set. Write a one or more pages report to summarize your algorithms, results and findings.

3. Please structure your source code in a professional data science coding style and framework like https://github.com/jc-audet/WOODS or https://github.com/timeseriesAI/tsai/tree/main/tsai.

****
* Appendix *
****

1. Time label
The functon epoch_time_local() will convert the Time (e.g., epoch) to a readable local timestamp string such as 2022-02-05T10:10:00.000. Note that this data set includes the data of 75 subjects, where the data of different day is from different subject. In some cases, there were two subjects test on the same day, where one is in the morning and the other is in the afternoon. The data was collected in the local time zone America/New_York.

```
def epoch_time_local(epoch, zone="America/New_York"):
   local_tz = pytz.timezone(zone)
   time =
datetime.fromtimestamp(epoch).astimezone(local_tz).strftime("%Y-%m-%dT%H:%M:%S.%f")
   return time
```

2. ID label
The functon int_to_mac() will convert the integer ID (e.g., macint) to the MAC address string, which is one of the four MAC addresses below: (H1) b8:27:eb:6c:6e:22 (H2) b8:27:eb:5b:35:37 (F1) b8:27:eb:23:4b:2b (F2) b8:27:eb:80:1c:cf, where H1 and H2 are under the same hospital bed, F1 and F2 are under the same family bed.

```
def int_to_mac(macint):
   newint = int(macint)
   return ':'.join(['{}{}'.format(a, b)
                for a, b
                in zip(*[iter('{:012x}'.format(newint))]*2)])
```