xyzab Data Analysis Task, Report

This document would provide code explanation as well as the summary of analysis on each person for each window. You are encouraged to read the corresponding python notebook side by side to see the complete code in action live too, and the data involved in more detail.

- **Code Explanation:**

This section would contain the code explanation in a manner as succinct as possible, and overlook the redundant/non-important parts. Again, you are encouraged to check the python notebook for a complete view.

```
import pandas as pd
data1 = pd.read_csv('csv_1.csv')
data1['Time'] = pd.to_datetime(data1['TimeStamp'],unit='s')
# Human Readable conversion of UNIX timestamp
cols = list(data1.columns.values)
data1 = data1[[cols[0]] + [cols[-1]] + cols[1:3]]
```

This section deals with the loading of the csv files to be able to be manipulated with python for our uses. The first line simply declares that the library **`pandas`** is being imported with an alias `pd`. We then use this very library to load our csv file one though the `pd.read_csv('csv_1.csv')` statement and assign it the `data1` identifier for our convenience.

Now the csv file had UNIX epoch timestamps which are neither user readable, nor are particularly nice to look at on graphs, so we converted them to human readable timestamps though the `pd.to_datetime` statement, and then through the ending two lines, we make another column named Time with the human readable time and arrange the column to be just besides the UNIX timestamp columns.

Our dataframe now looks like this :

	TimeStamp	Time	HeartRate	BreathRate
0	1599419501	2020-09-06 19:11:41	92.38	10.30
1	1599419531	2020-09-06 19:12:11	92.38	16.97
2	1599419561	2020-09-06 19:12:41	73.50	16.97
3	1599419591	2020-09-06 19:13:11	76.38	16.97
4	1599419621	2020-09-06 19:13:41	58.93	16.97

Now, as per the given instructions, we needed to have a 1 hour non overlapping window for each person's record, and this is achieved through the following statements:

```
data_hrt0 = data1[['Time'] + ['HeartRate']]
data_hrt = data_hrt0.iloc[:121]
```

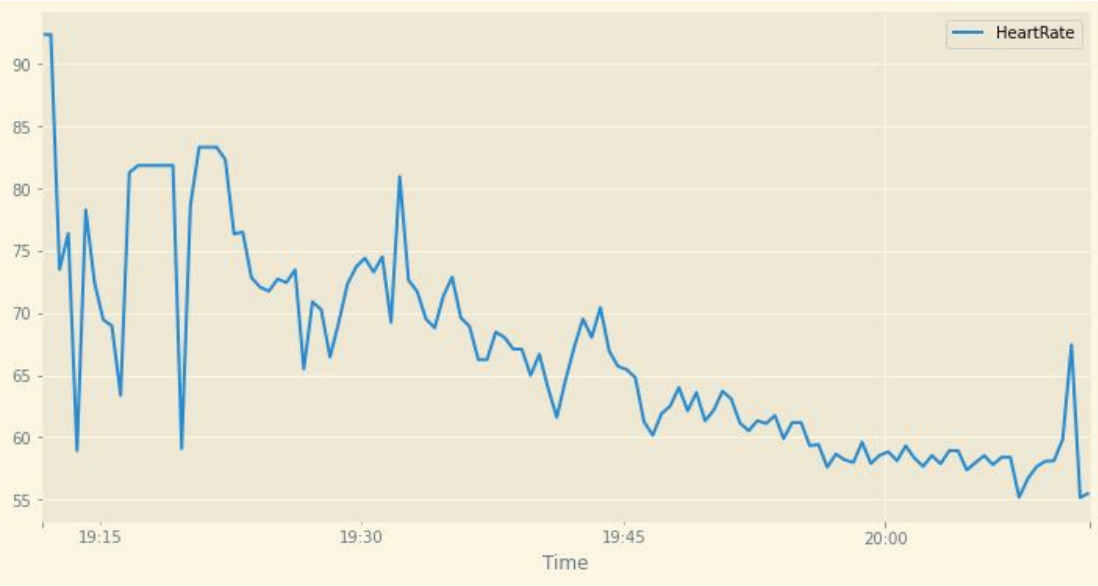
We made another dataframe for heart analysis, and took the columns **Time** and **HeartRate** into consideration. The second statement uses `iloc` to separate out 120 columns as per our need of 1 hour windows.

```
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import numpy as np
%matplotlib inline
plt.style.use('Solarize_Light2')
plt.rcParams["figure.figsize"] = (12,6)
```

Next come the statements above. Here, as we see we need the graphs, we import the library **matplotlib** which is used for plotting graphs. We also import NumPy for some operations just in case. To make inline graphs, we put the statement `%matplotlib inline` and then we choose the Solarized theme for our graphs and set the size of the plots to be 12x6 inches.
Next comes the graphing part.

```
fig0 = data_hrt.plot(x = "Time", y = "HeartRate", label='HeartRate')
```

This statement gives us the following graph :



However, as we can see, there are some steep cliffs here. To make this more normal looking, and to get the overall trend of the quantity being considered, we use the rolling window function implemented in pandas itself, and create another column titled **SMA** through the following statement:

```
data_hrt['SMA'] = data_hrt.iloc[:,1].rolling(window=4).mean()
```

We then plot a graph using the new moving average we got and overlay it on the original plot using the following statements:

```
ax = data_hrt.plot(x = "Time", y = "HeartRate", label='HeartRate')
fig1 = data_hrt.plot(x = "Time", y = "SMA", label='Moving Average of 4 for HeartRate', color = 'red', ax = ax)
```

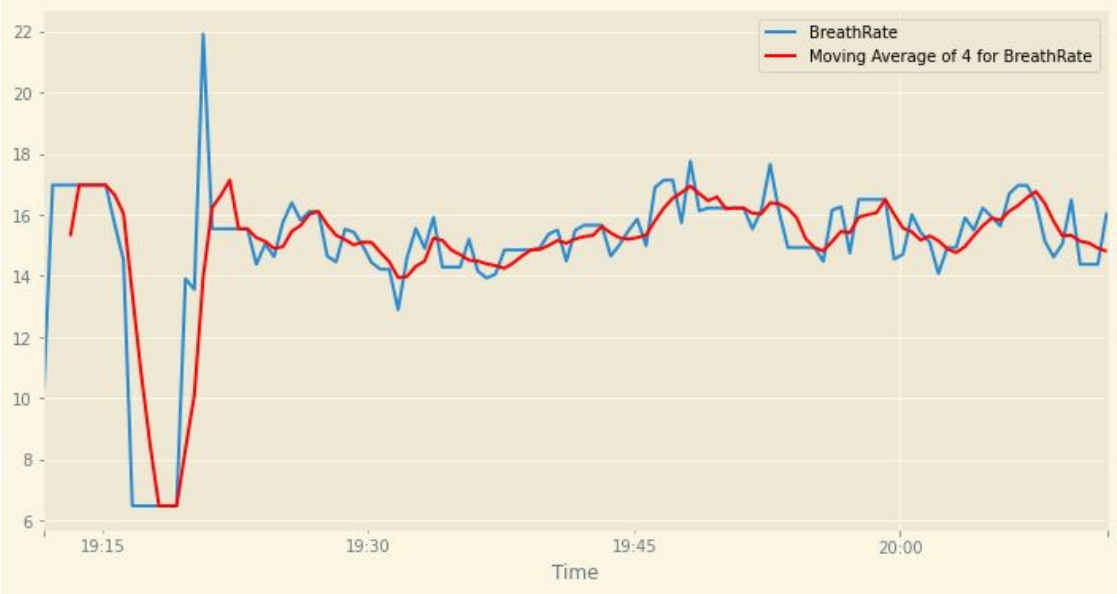
We thus obtain the following plot :

Now while it was apparent from the original plot that the trend here is downwards, it is now even more apparent thanks to our new SMA feature. This also removed the spikes and cliffs from the original graph.

Now, while the above was for the heart rate of the person whose data we are considering, we now do it for the Breath Rate too.

```
data_brt = data1[['Time'] + ['BreathRate']]
data_brt = data_brt.iloc[:121]
data_brt['SMA'] = data_brt.iloc[:,1].rolling(window=4).mean()
ax = data_brt.plot(x = "Time", y = "BreathRate", label='BreathRate')
fig1 = data_brt.plot(x = "Time", y = "SMA", label='Moving Average of 4 for
BreathRate', color = 'red', ax = ax)
```

Similar code as above, and we obtain the following plot for breath rate:



Note that this again removed the sheer spike via the SMAs.
Now, this same procedure is repeated for each window for this person, and then the entire procedure is cycled for all of the 10 people in question.

To make the process more simpler, we choose to do the windowing part of both heart rate and breathing rate combined in one block, like this :

```
# For second window
data_2= data1.iloc[121:241]
#heartrate window 2
fig10 = data_2.plot(x = "Time", y = "HeartRate", label='HeartRate')
# Large spikes observed. Correcting.
data_2['SMA'] = data_2.iloc[:,2].rolling(window=4).mean()
ax = data_2.plot(x = "Time", y = "HeartRate", label='HeartRate')
```

```
fig11 = data_2.plot(x = "Time", y = "SMA", label='Moving Average of 4 for
HeartRate', color = 'red', ax = ax)
#breathrate window 2
fig12 = data_2.plot(x = "Time", y = "BreathRate", label='BreathRate')
data_2['SMA2'] = data_2.iloc[:,3].rolling(window=4).mean()
ax = data_2.plot(x = "Time", y = "BreathRate", label='BreathRate')
fig13 = data_2.plot(x = "Time", y = "SMA2", label='Moving Average of 4 for
BreathRate', color = 'red', ax = ax)
```

After this process is completed for the entire duration of the sleep, we finally generate some statistics about the full sleep through the following command, while also generating a copy of the CSV with the modifications done by the program with the set of operations we did on it :

```
data1.to_csv('csv1_modified.csv', index = False)
# Saving the modified csv to the keep the original data intact
data1.describe()
```

Thus we generate the following statistics:

	TimeStamp	HeartRate	BreathRate
count	9.840000e+02	984.000000	984.000000
mean	1.599434e+09	57.975315	15.707612
std	8.526019e+03	5.936772	2.862147
min	1.599420e+09	49.150000	6.490000
25%	1.599427e+09	54.260000	14.325000
50%	1.599434e+09	55.970000	15.270000
75%	1.599442e+09	59.805000	16.410000
max	1.599449e+09	92.380000	31.120000

Again, this process is done again for all the people in consideration.It should be noted that the process of the generating stats can be done for each window too, should it be desired.

- **Summary of the trends for each of the people involved:**

Again, you are advised to see the python notebooks to see the visualizations for each window.

Person 1:

Window #	HR Trend	BR Trend
1	Downward	Stable (Starts Erratic)
2	Stable	Erratic with high variations
3	Mostly Stable	Highly Erratic
4	Overall Stable	Highly Erratic
5	Increasing	Erratic
6	Erratic	Erratic
7	Stable	Erratic
8	Stable	Erratic
9	Erratic	Erratic

This person has an erratic pattern in both parameters several times, thus if I had to assign a sleep score to this person, I wouldn't give it a high one.

Person 2:

Window #	HR Trend	BR Trend
1	Downward	Erratic
2	Downward	Stable
3	Stable	Erratic
4	Stable	Erratic
5	Downward	Stable
6	Stable	Stable
7	Stable	Stable
8	Stable	Stable
9	Stable	Stable

While this person has an erratic pattern in BR several times, they only exist towards the beginning. As the person sleeps for more durations, they manage to attain a stable state which I think is a good thing.

Person 3:

Window #	HR Trend	BR Trend
1	Increasing	Erratic
2	Stable	Erratic
3	Stable	Erratic
4	Stable	Downward
5	Stable	Stable, Erratic towards end
6	Stable	Stable, Erratic towards start
7	Stable	Stable
8	Downwards	Erratic
9	Erratic	Erratic

Observations:

1. The erraticness in breathing in window 5 and 6 seems to be a continuation.
2. Breathing remains mostly erratic
3. Heart rates are surprisingly stable in comparison.

Person 4:

Window #	HR Trend	BR Trend
1	Erratic, then stable	Erratic, then stable
2	Stable	Erratic
3	Stable	Erratic
4	Erratic	Stable
5	Stable	Stable
6	Erratic	Erratic

Observations:

1. The ending has lots of erratic patterns
2. The duration is short
3. Not a very high score for sleep

Person 5:

Window #	HR Trend	BR Trend
1	Rising	Mostly Stable
2	Erratic	Stable
3	Downward	Stable
4	Stable	Stable
5	Stable	Stable
6	Stable	Stable

Observations:

There seem to be some erroneous values here, despite it, the person gets a more stable sleep as it progresses. The duration of the sleep does seem short, though.

Person 6:

Window #	HR Trend	BR Trend
1	Stable	Erratic, but decreasing
2	Mostly Stable	Erratic
3	Erratic	Erratic
4	Increasing then decreasing	Erratic
5	Decreasing	Mostly Stable
6	Decreasing	Stable

Observations:

Several Erratic instances, low time of sleep give this one a low score.

Person 7:

Window #	HR Trend	BR Trend
1	Stable	Stable
2	Stable	Stable
3	Stable	Stable
4	Stable	Stable
5	Stable	Erratic
6	Stable	Mostly Stable
7	Stable	Mostly Stable
8	Stable	Stable

Observations:

The device seems faulty here. There is a lack of data in some instances, while there is a very suspicious straight lines in HR. If the device is working correctly, then this would have earned a very high score due to the stable scores as well as a perfect length of sleep.

Person 8:

Window #	HR Trend	BR Trend
1	Stable	Erratic, abnormal
2	Stable, increasing towards the end	Stable
3	Stable	Stable
4	Erratic	Erratic
5	Stable	Stable
6	Erratic	Erratic

Observations:

In window 1 the person has an abnormal rate of breathing. The overall HR is also quite high, and there are several erratic instances, aside from the duration being very small. Not a high score.

Person 9:

Window #	HR Trend	BR Trend
1	Erratic	Erratic
2	Mostly Stable	Erratic
3	Erratic	Erratic
4	Erratic	Mostly Stable, increasing towards the end
5	Erratic	Erratic
6	Erratic	Erratic
7	Erratic	Somewhat stable
8	Erratic	Somewhat stable
9	Erratic	Stable

Observations:

The person shows a very erratic HR with a very high range and variance. The BR is also replicating mostly the same behaviour. Not a good score.

Person 10:

Window #	HR Trend	BR Trend
1	Stable	Stable
2	Erratic	Stable
3	Stable	Stable
4	Downward	Stable
5	Mostly Stable	Stable
6	Erratic	Erratic
7	Erratic, Upward	Erratic
8	Stable	Slightly Downward
9	Stable	Stable
10	Erratic	Erratic

Observations:

The person’s HR is in higher ranges consistently, and keeps changing from stable to erratic and vice versa. Breathing is uneven too. Not a good score.

- **Further Prospects**

This data can further be used in several ways, for example, one can generate a scatter plot with this and compare the ranges as well as classify the people in groups like, having high HR, low HR, high BR and low BR. The sleep duration can also be compared, and the sleep score factor can be used to compare against other people to encourage more healthier sleep habits.

For example, something like this can be done. The code for this is fairly basic and included with the zip file with the name ‘overall_analysis’.

Now while this compares only the overall mean. Similar statistics can be made for other measures too, and users can compare their daily statistics through this. We can also classify through graphs of these types and find which users of the product (given their consent) are having a good sleep. Similarly, statistics like these can also be used to promote the product if more users adhere to better sleeping practices after checking statistics like these.