CSE 572: DATA MINING Fall 2017
ASSIGNMENT 3
Group: SuperBug

Task 1 Explanation:

In task 1, we have developed a Matlab code using the annotations and the data of all 32 groups, to build two csv files namely: EatingAction.csv and NonEatingAction.csv.
Steps performed:

- Cleaning the data, standardizing the format.
- Extracting the data of each user and map based on the annotation files to classify the sensor data (IMU and EMG) whether it is an Eating Action or Non-Eating Action.
- Scaling Factor: Since the number of frames of the video and the number of observations in the raw data are different, we decided to make use of a scaling factor which can map the data correctly corresponding to the video frames. After analyzing the raw data of about 8 groups, it was deduced that the IMU data for spoon was sampled at 30 Hz, and EMG data was 60Hz. Similarly, for Fork, the sampling rate for IMU, was 50Hz and EMG was 100Hz. All frequencies are approximated. So accordingly, we had 4 scaling factors, we mapped the dataset.
- The rows in the CSV files denote the sensor data of each eating action. Example:
    - Row1: EatingAction1: Accelerometer X,
    - Row2: EatingAction1: Accelerometer Y,
    - Row3: EatingAction1: Accelerometer Z and so on. (18 sensor data for Each Eating/ Non-Eating Action)
- Zero padding has been added to the remaining entries to get a Uniform Matrix.

Please find the final CSVs named: EatingAction.csv and NonEatingAction.csv.

Task 2: Feature extraction:

In the second task, we have applied multiple feature extraction techniques on the Data accumulated that can denote distinction between Eating and Non-Eating actions on multiple user dataset. We had matrices named: EatingAction.csv and NonEatingAction.csv which consisted of all the data of the sensors from 32 users. We applied various feature extraction techniques to find the best possible features on the extracted dataset. We plotted 90 graphs (18 sensors x 5 Features) for Eating and Non-Eating actions to visualize which sensor data can be used to identify Eating and Non-Eating Action distinctly. We used the original matrix without padding to compute the following features.

The following five features extraction techniques were used:
1. Variance
2. Root Mean Square
3. Kurtosis
4. Skewness
5. Standard Deviation

By Visualizing the graphs, we found as the good features.
Explanation of feature extraction techniques used:

1. Write an explanation on how the feature is extracted.
We used Fast Fourier Transformation (FFT) on the raw data matix obtained in task1. Later, we used a Matlab's inbuilt functions for each feature extraction technique, which calculated the value of that feature on the FFT-transformed data.

2. Write an intuition on why you use such a feature.

1) **Variance**: Variance gives the spread of the data. So we thought variance was a good measure as it can capture the variability among the sensor data while performing the eating and non eating actions.
2) **Root Mean Square**: RMS calculates the magnitude of action performed by the user. If RMS is used, it can provide continuous periodic wavelengths which repeats itself. Since we perform eating actions at a periodic time (Assumption), we decided to use RMS as the feature.
3) 3. **Kurtosis**: Since we have the data for 32 users, there is a high chance that the eating actions (and non eating) actions are going to have a normal distribution such that we would receive the peak of eating action much before the peak of non eating action. This means that there is a good chance of non eating action forming a tail at the end of its time period.
4) **Skewness** : Skewness defines the measure of the similarity so assuming that all the eating actions are going to have a similar period they should be similar to each other. Also, the usual wait time between the end of previous eating action and the start of the next eating action (non eating) will be similar too for a user. Therefore, we selected skewness as a feature extraction technique.
5) **Standard Deviation**: Since, standard deviation is a nonlinear function of variance. We chose standard deviation because if variance failed to capture the variability (i.e. data is non linearly dependent) then standard deviation can be thought of a good measure to extract/transform data.

3. Write a Matlab code to extract that feature from each time series stored in the csv files created in task 1.

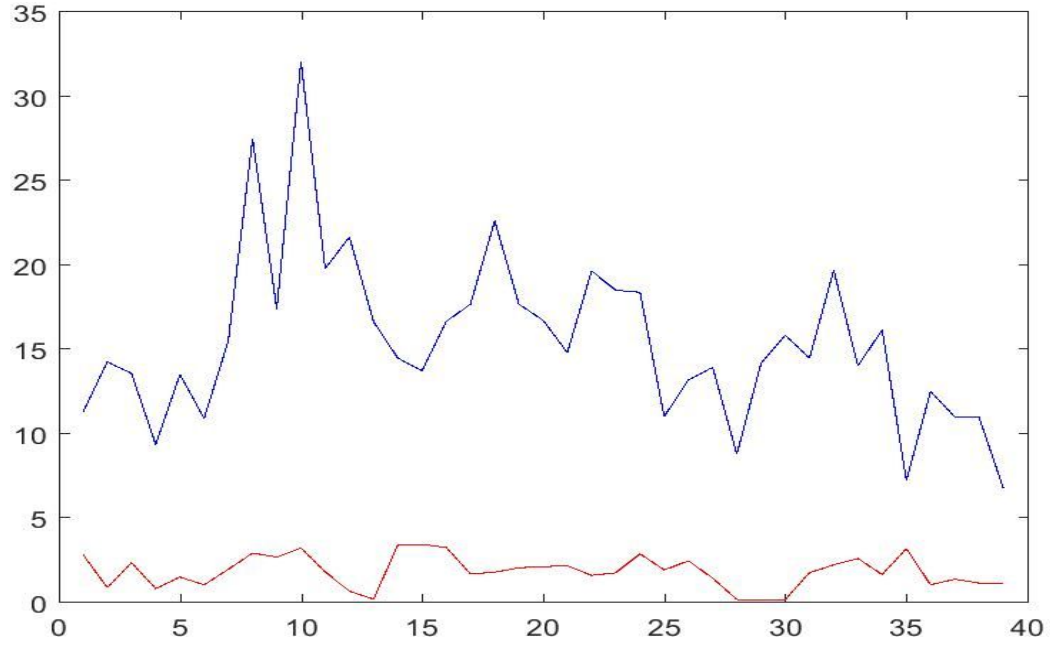
- Code has been submitted in the zip file with the name: Assignment3.m

4. Generate plots: i) features extracted from all eating actions, and ii) features extracted from all non-eating actions. For multiple eating or non-eating actions you can choose to overlap the plots. This will give you a better idea of potential patterns in the features.

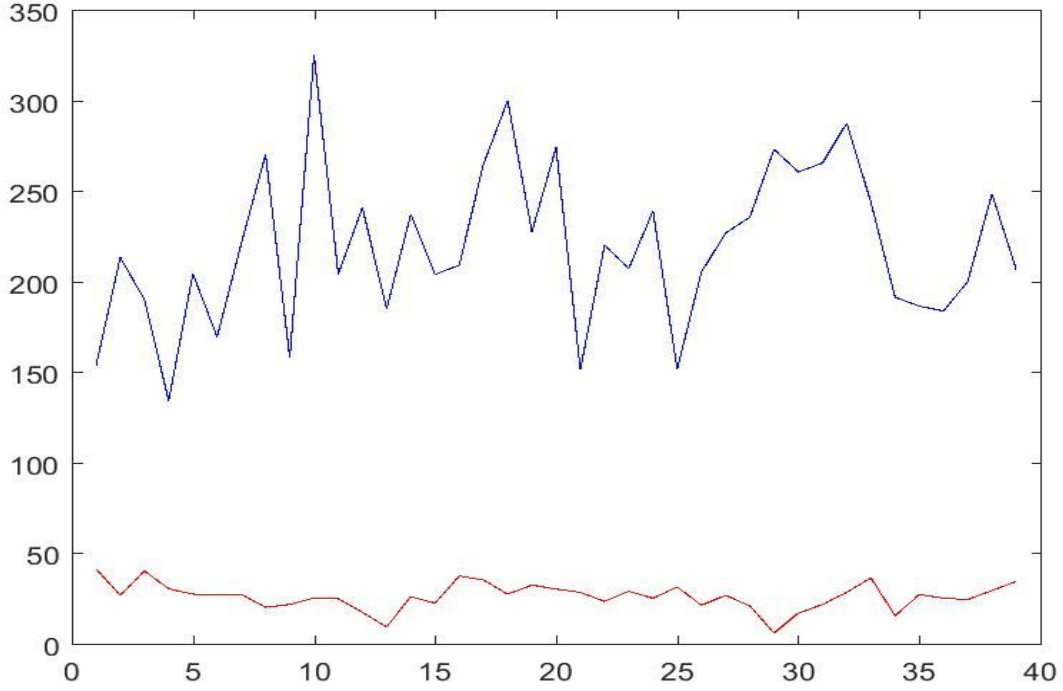
SINGLE USER GRAPHS :
To analyse the best features, we first ran feature extraction techniques to small set of data, and later performed the same action on data of all the users. This consist of running the feature extraction techniques on multiple eating and non eating actions of a single user. As we found the following features in single user data as the best of all, so we used this features for all the users.

In all the graphs below:
Eating Action is denoted by **Red Line.**
Non Eating Action is denoted by **Blue Line.**

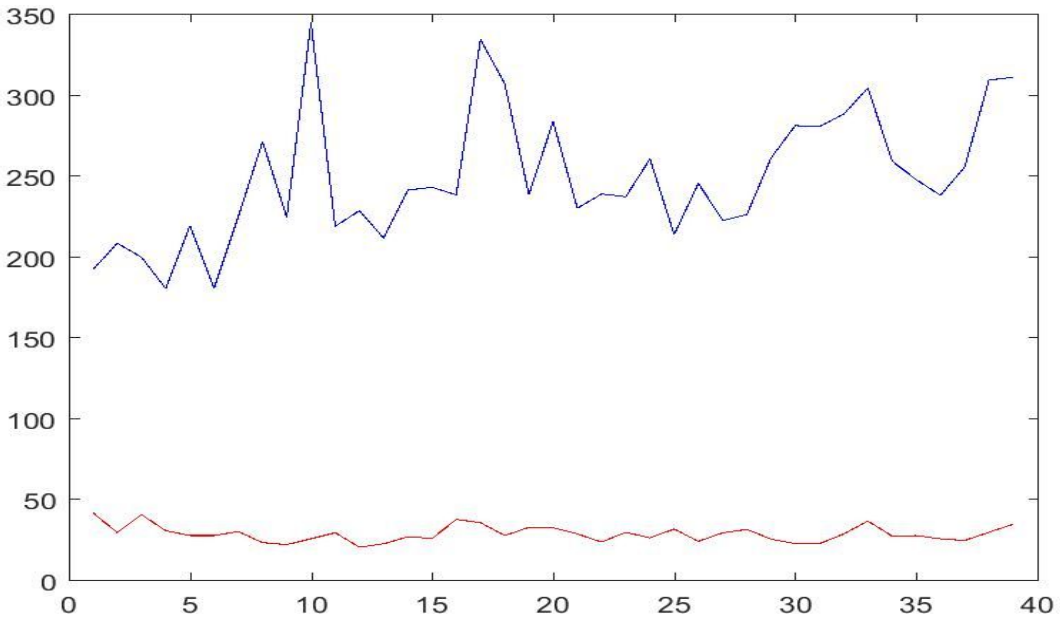
1. **Variance (Gyroscope Y)**



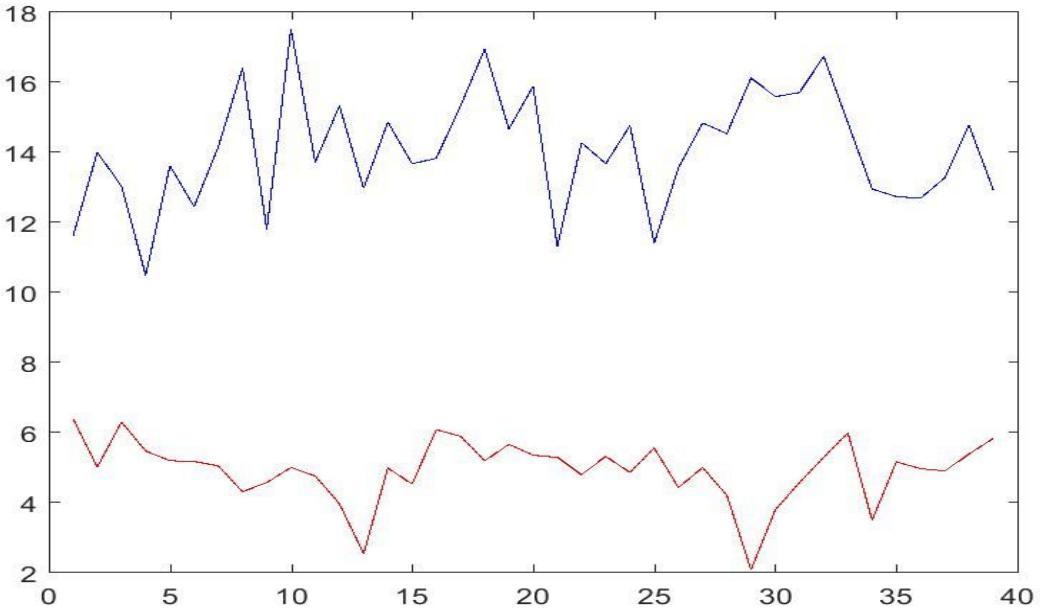
2. A. Kurtosis (Gyroscope Y)



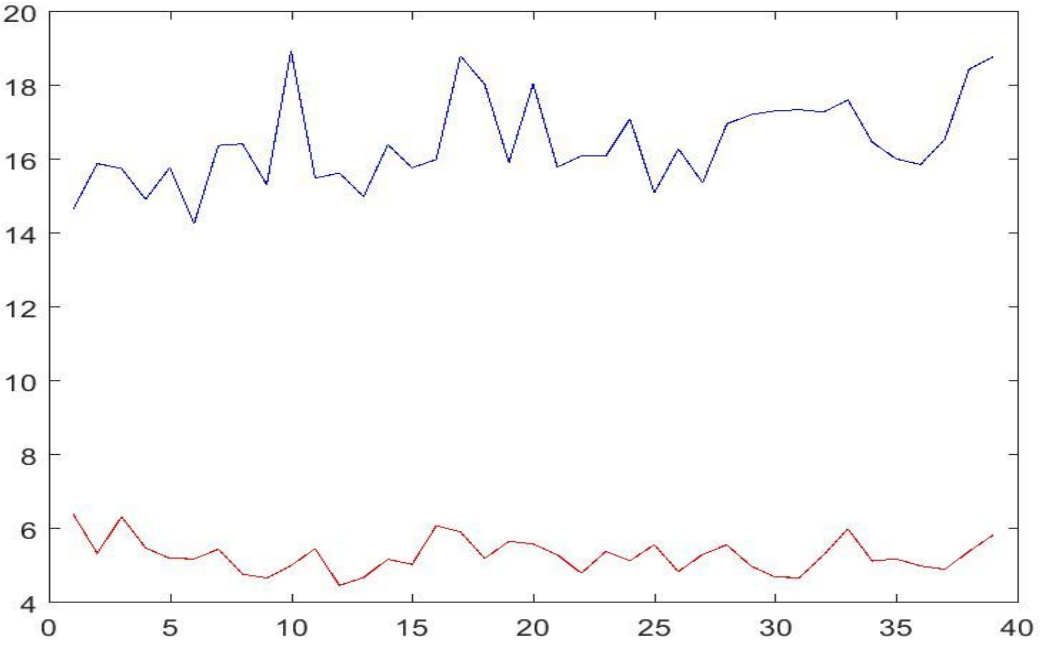
2. A. Kurtosis (Accelerometer Y)



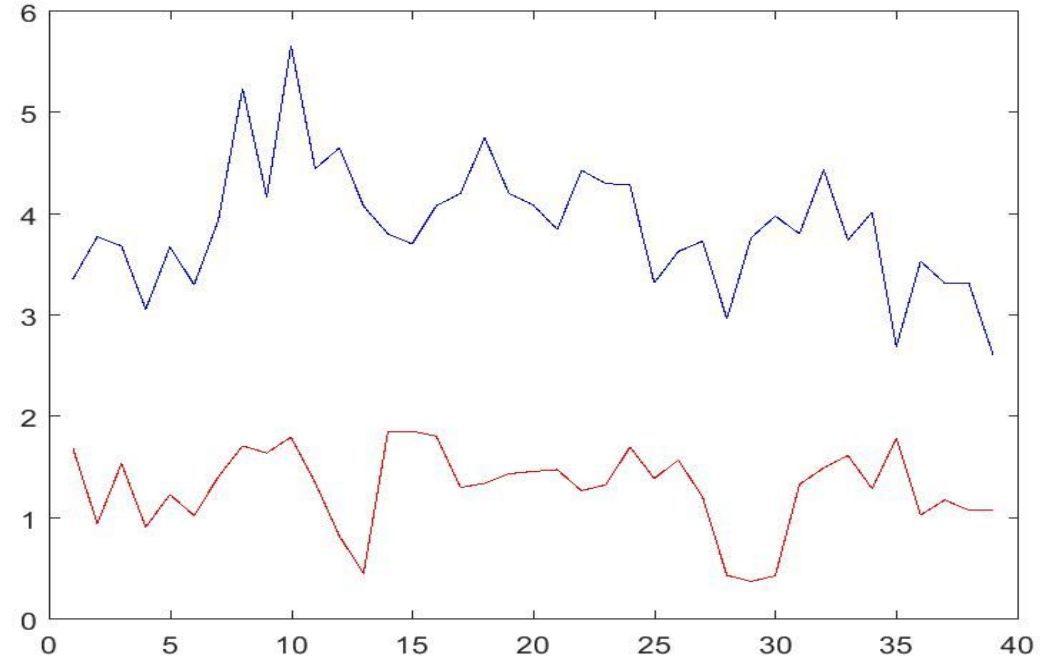
3. Skewness (Gyroscope Y)



4. B. Skewness (Gyroscope X)

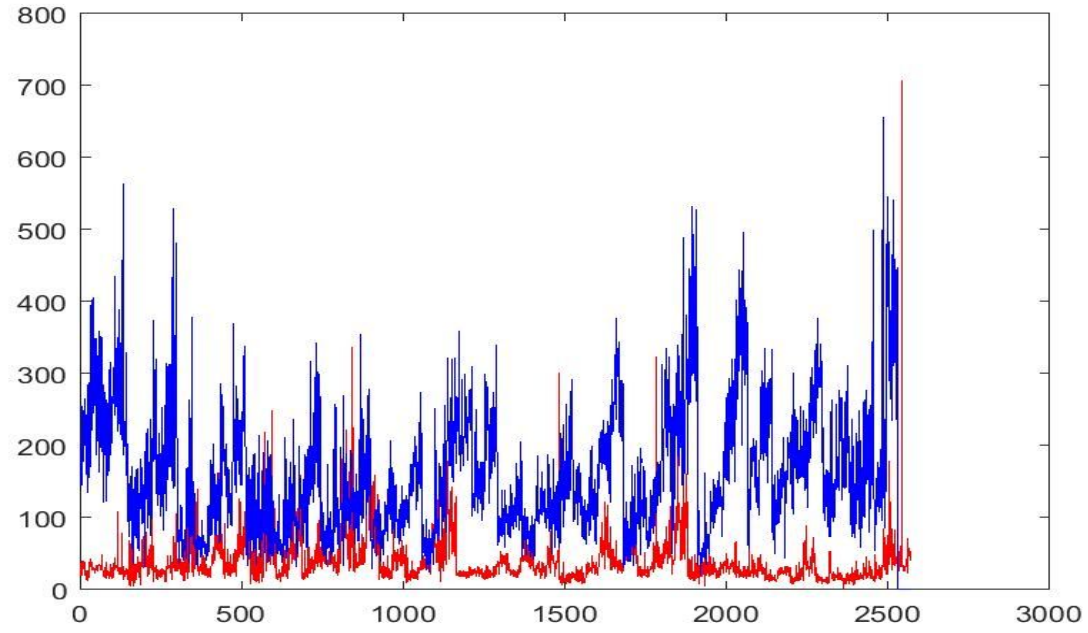


5. Standard Deviation (Gyroscope Y)

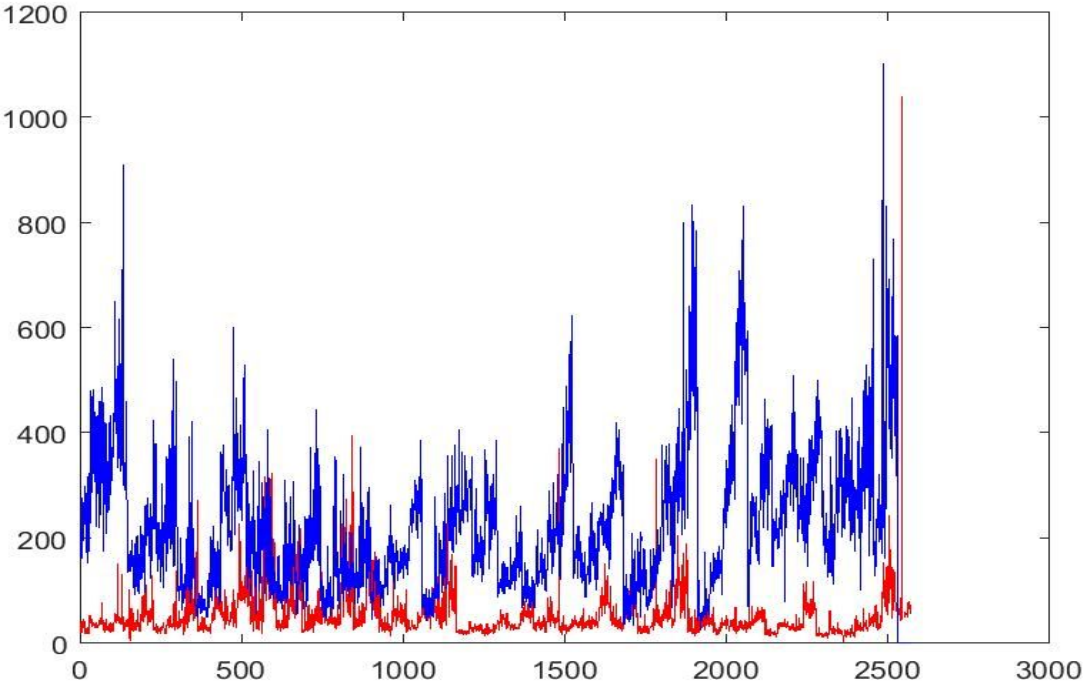


ALL USER GRAPHS (Combined)

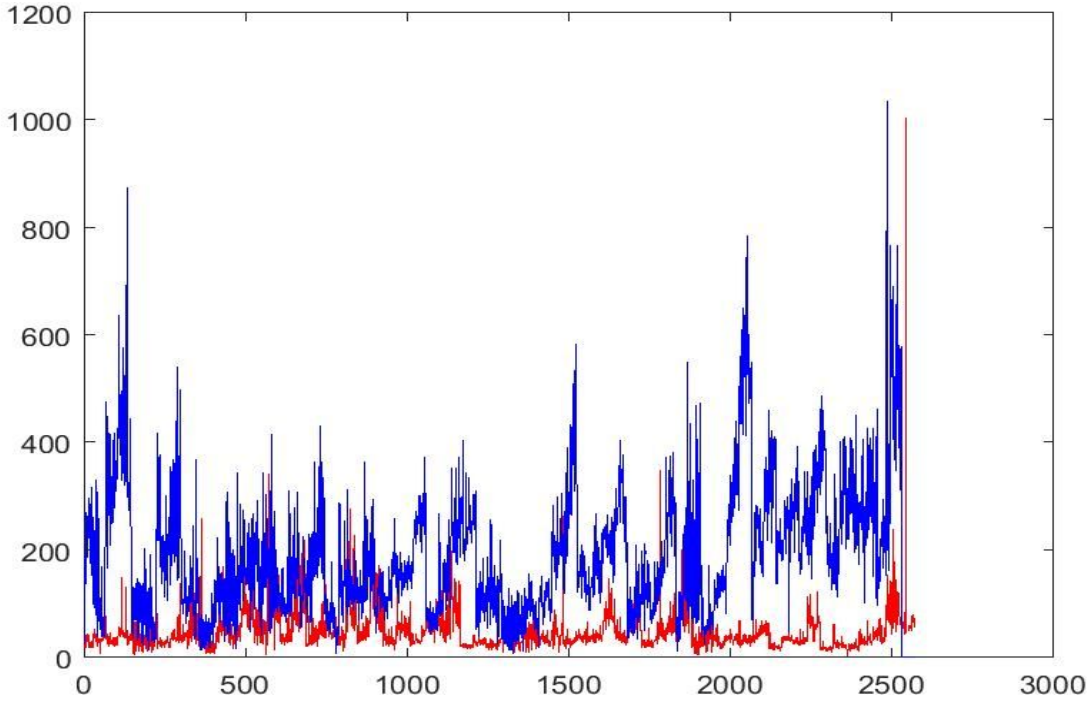
1. Variance (Gyroscope Y)



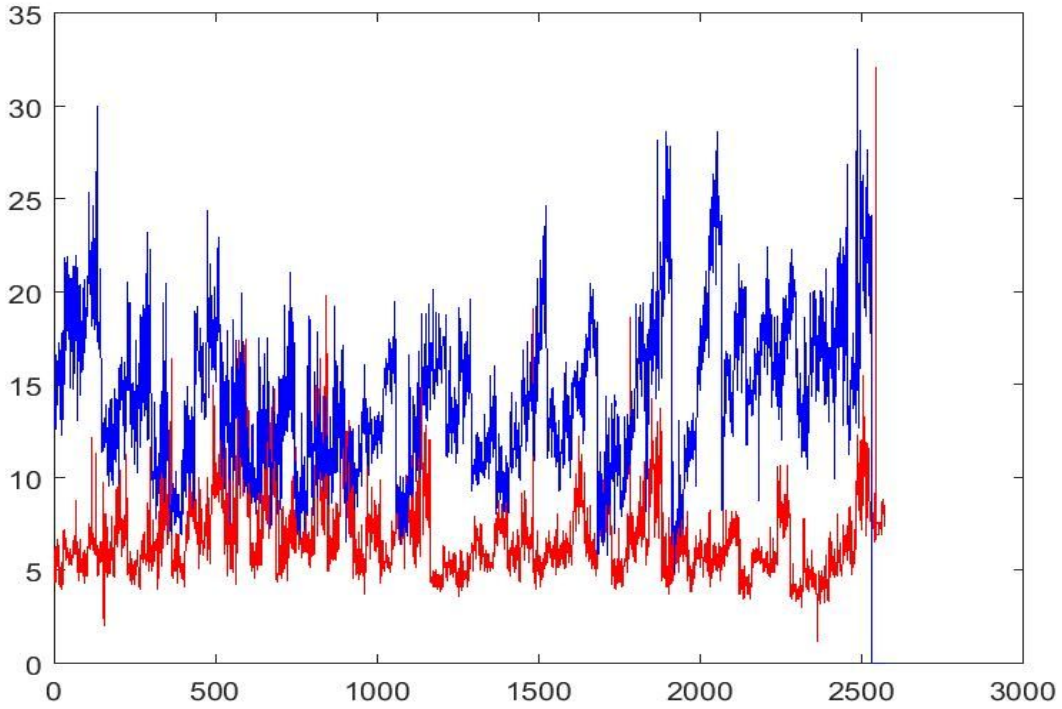
2. A. Kurtosis (Gyroscope Y)



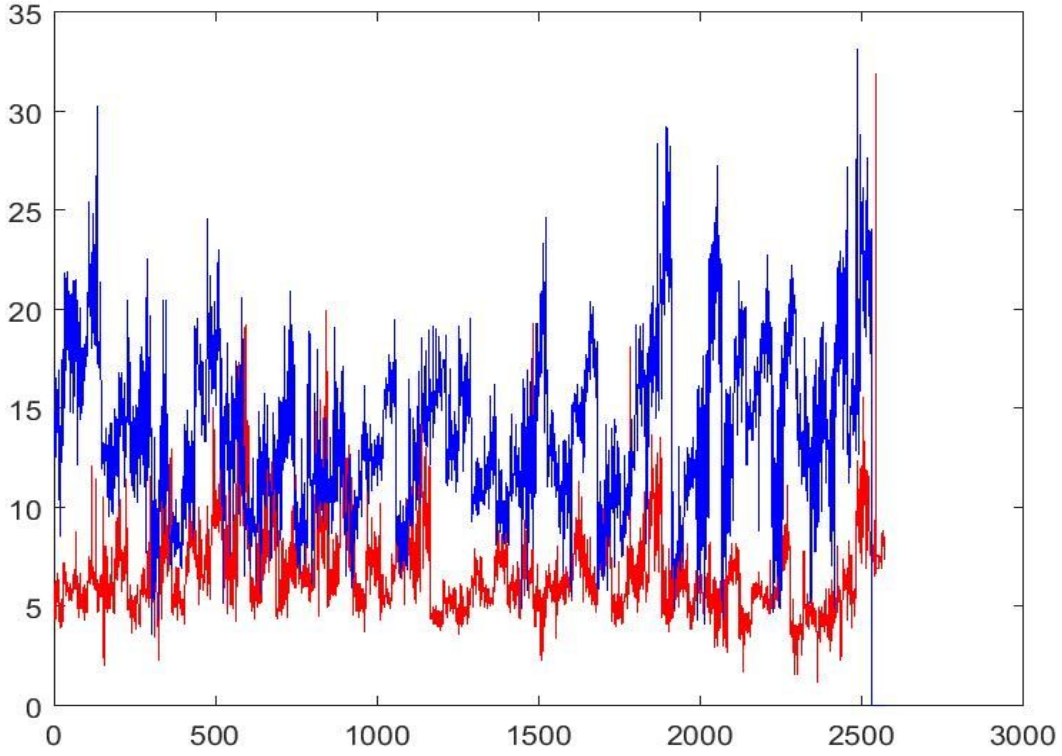
2. B. Kurtosis (Accelerometer Y)



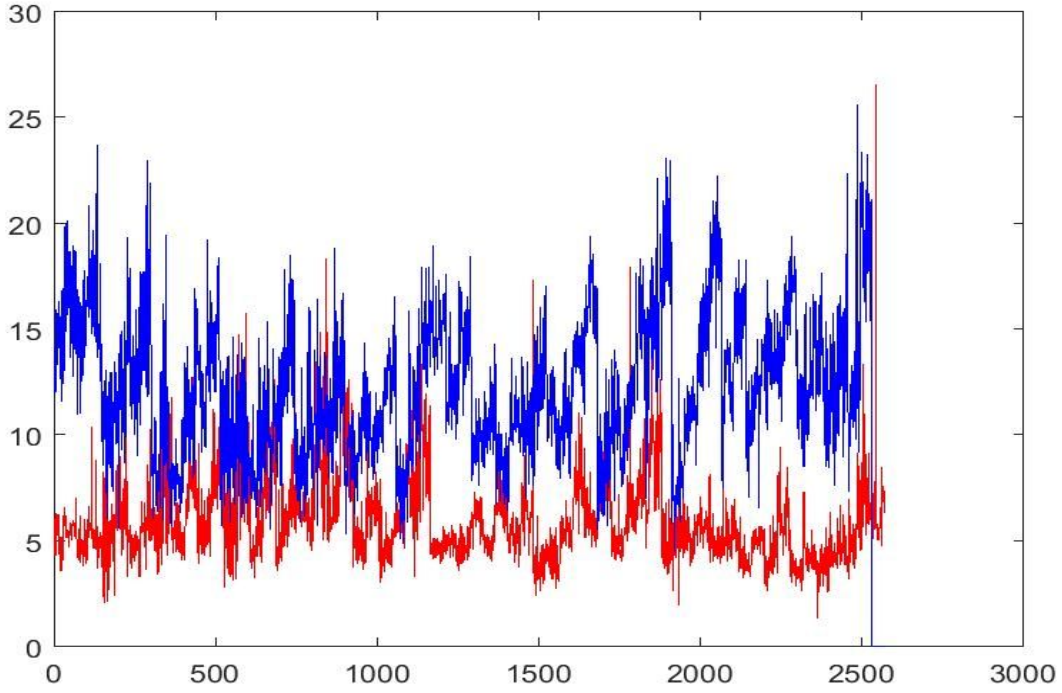
3. A. Skewness (Gyroscope Y)



3. B. Skewness (Gyroscope X)



4. Standard Deviation (Gyroscope Y)



5. Discuss whether your initial intuition about the features that you selected holds true or not.

The above graphs are the best distinctions we could obtain between the eating and non eating actions after applying the feature extraction techniques on the FFT transformed data. The explanations for the following features are given below. It can be seen that the sensor gyroscope in Y direction proved to be of reasonable importance when we are trying to differentiate between eating and non eating action.

1. Variance: The initial intuition was seen among sensors with respect to variance however the variability in most of the sensors was overlapping. They gyroscope Y showed a better variance among all others. Moreover, we assume that the data of 32 users is non uniform and hence there was no clear distinction among the eating and non eating actions. From the above graphs you can see that variance is a pretty good measure when applied on data of one user.

2. Root Mean Square: RMS didn't prove to be a good feature and it didn't correspond to our intuition. This can be attributed to the fact that each user will take different amounts of time to complete either of the eating actions and hence the different magnitudes which we assumed to be of similar values. Hence, RMS failed.

3. Kurtosis: RMS failed because it assumed that different users will have similar time periods and hence similar magnitude. However, the feature kurtosis shined here because even though different users had different time periods their combined data formed a normal distribution for eating as well as non eating action. The two normal distribution curves can be differently distinguished for the two actions and hence you can see a better distinction.

4. Skewness: Skewness is the measure of symmetry in the data. Since we got normal distributions from the kurtosis feature, we assumed we will get good results from skewness, which we got to some extent. But we are unsure, if the results we got were because of the intuition or not.

5. Standard Deviation: It did not prove to be of much importance because interpreting its results, we can see that the sensors are not non linearly related to each other (it might be related in some other way but not in non-linear way). Our main intuition to use standard deviation was to see whether there is any non linear dependency.

Task 3: Feature Selection

PCA is a dimensionality reduction technique which can be applied to reduce the total number of dimensions. We performed the PCA using the inbuilt Matlab function and found the following results.

Subtask 1: Arranging the feature matrix
You know PCA only takes one matrix. How will you arrange all sensors and their corresponding features into a single matrix such that the eigenvectors of the covariance matrix directly makes sense to your data set? This means that if the PCA results gives you a eigenvector then the new feature matrix can be obtained by simply multiplying the eigenvector with the old feature matrix. (You might need two matrices corresponding to the two classes)
Write your logic of feature matrix arrangement.

When feature extraction techniques are applied to the raw data matrix we get 5 matrices of dimensions 40(actions) x 18(sensors). We get 5 matrices each for a feature extraction technique. Hence, at the end we will have 5 feature matrices which is going to be sent to the pca function to determine whether pca can make sense out of the data or not.

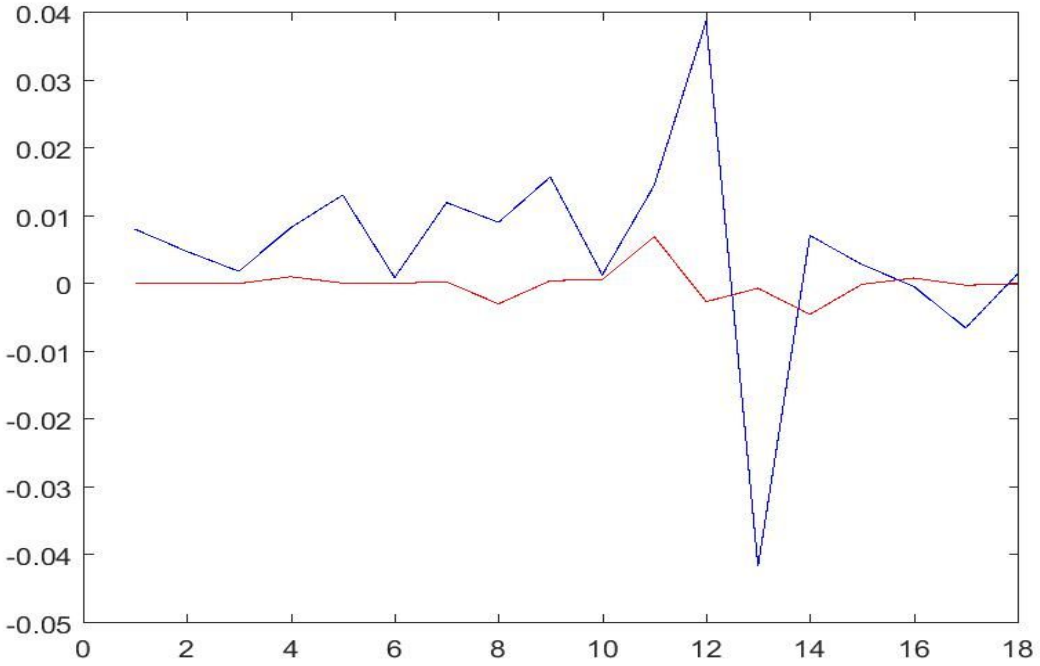
This technique is being used since there exist no method to perform PCA on distributed data tables, (heterogeneous sources of data), hence we have used a collective PCA approach in this case. Collective PCA runs across each feature extraction techniques final matrix and provides us with no. of eigenvectors which can be used to present that data. After performing CPCA, dimension compression occurs in each case.
[Reference:
[https://books.google.com/books?id=KhBuCQAAQBAJ&pg=PA452&lpg=PA452&dq=can+pca+have+hetrogenous+source+of+data&source=bl&ots=eKQhoeBiwZ&sig=sl8qupTr7iqLud9n_uo3RojJPXo&hl=en&sa=X&ved=0ahUKEwiq2baTmozXAhXGyFQKHT6ADocQ6AEIKjAA#v=onepage&q=can%20pca%20have%20hetrogenous%20source%20of%20data&f=false]

Subtask 2: Execution of PCA
Use Matlab's PCA function to run PCA on your feature matrix. Show all the eigenvectors in a plot.
Matlab provides inbuilt function to calculate the pca function on a matrix which returns coeff matrix (eigenvectors), score structure and latent (eigenvalues). This can be used to find out the required number of dimensions after PCA.

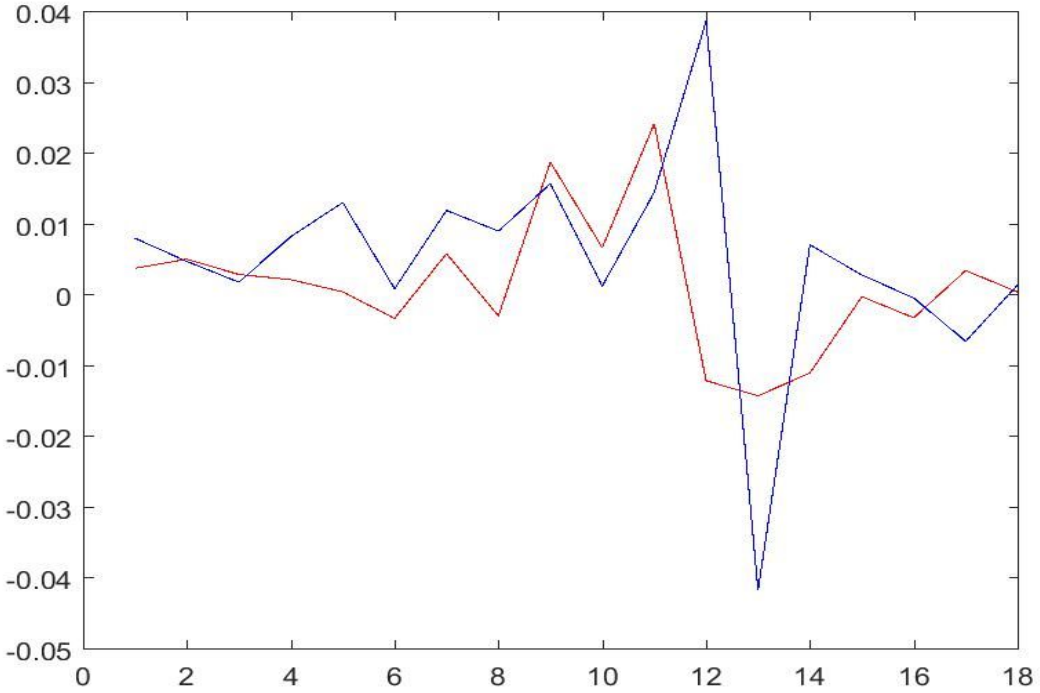
Eating Action is denoted by **Red Line.**
Non Eating Action is denoted by **Blue Line.**

Plots of EigenVectors:

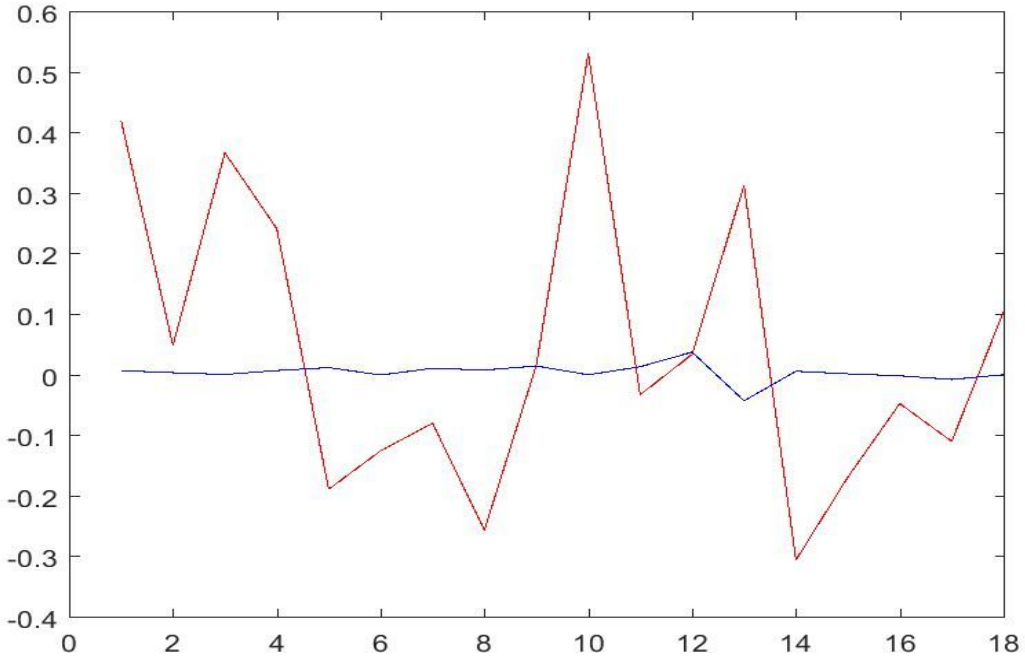
1. **Variance_EigenVectors**



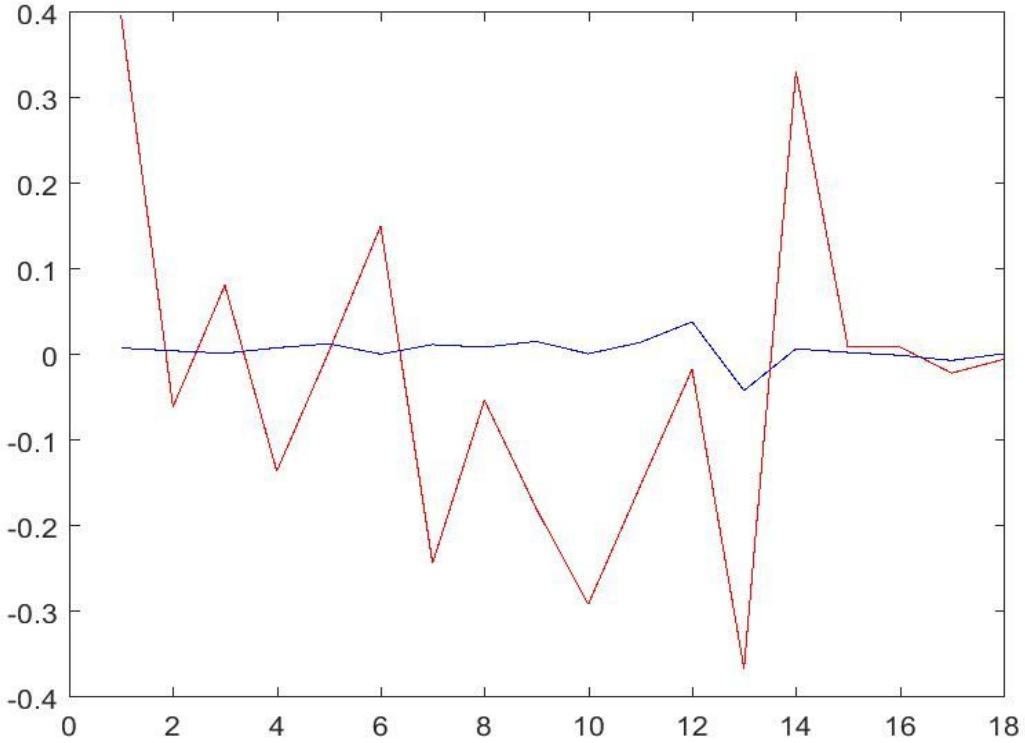
2 Rms_EigenVectors



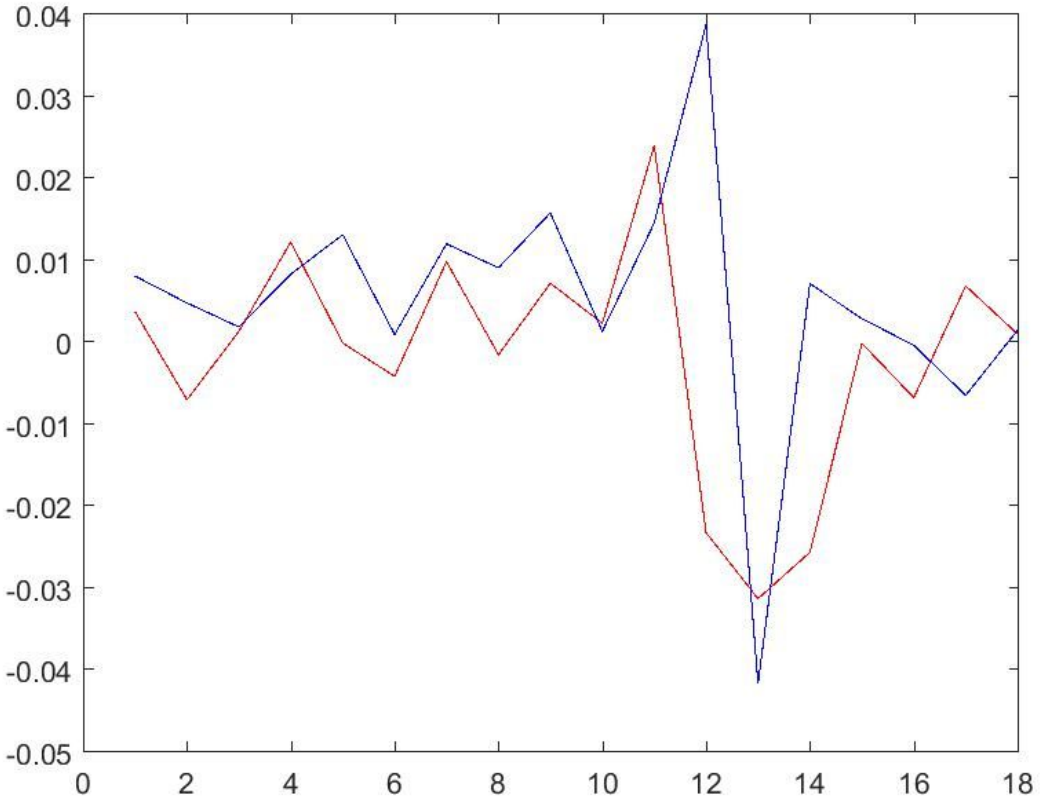
3. Kurtosis_EigenVectors



4. Skewness_EigenVectors



5. Standard Deviation_EigenVectors



Subtask 3: Make sense of the PCA eigenvectors

Write an explanation on the reason why the eigenvectors turned out the way they did.

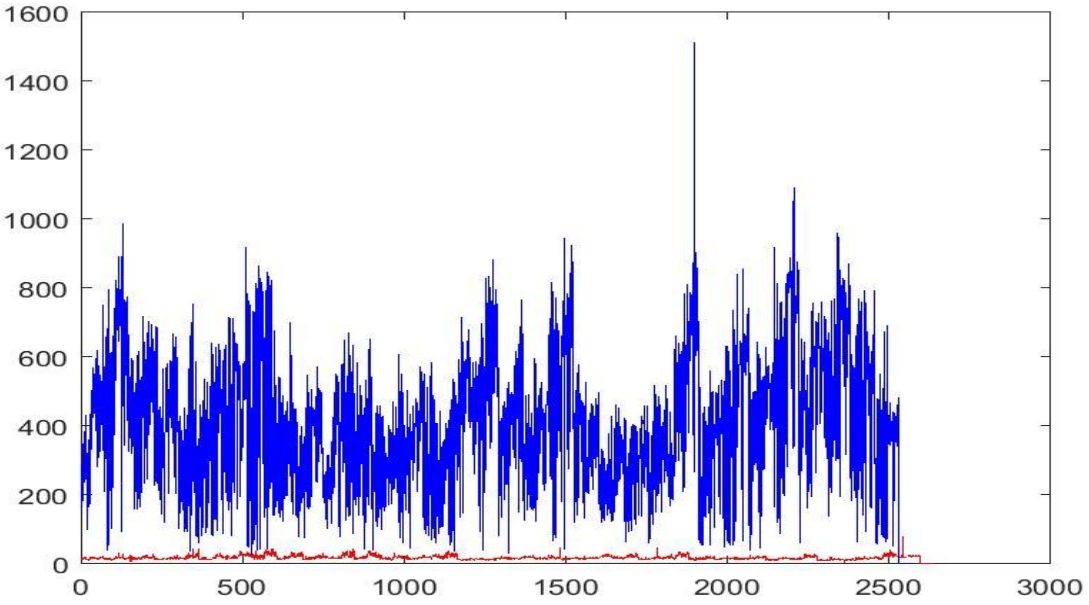
The eigenvectors represent the direction in which the data is most separated from each other among all the columns. It gives a value how correlated the any two columns are. The closer the value is to 1, the more correlated the columns are to each other. In the PCA that we applied, we observed that

Subtask 4: Results of PCA

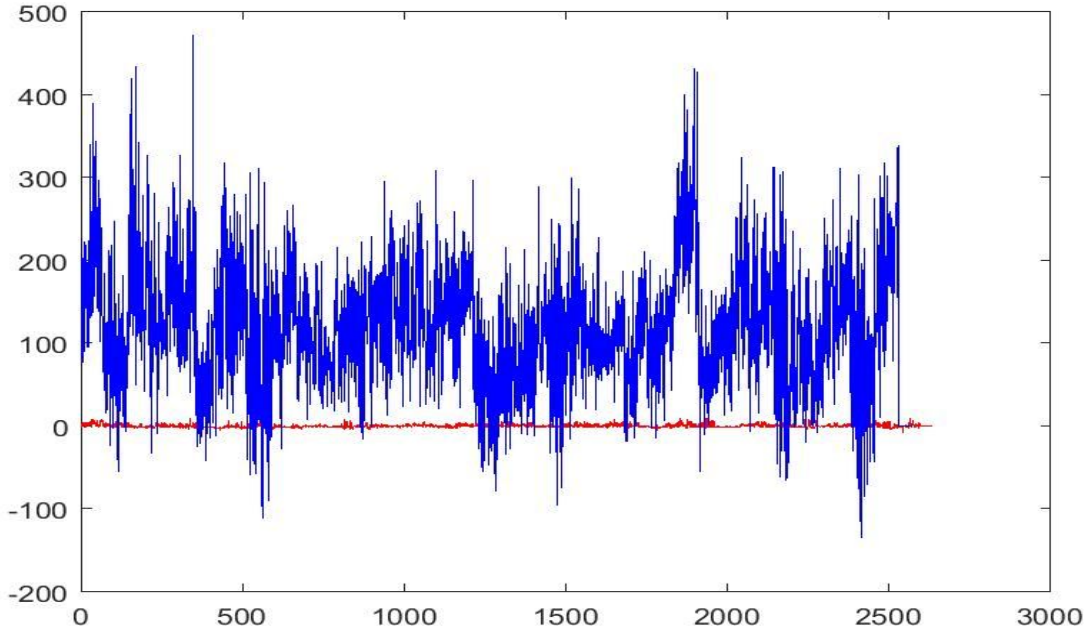
A new feature matrix was created by extracting features from all the eating and non eating actions by overlapping multiple earing and non eating actions of all the users.

Eating Action is denoted by **Red Line.**
Non Eating Action is denoted by **Blue Line.**

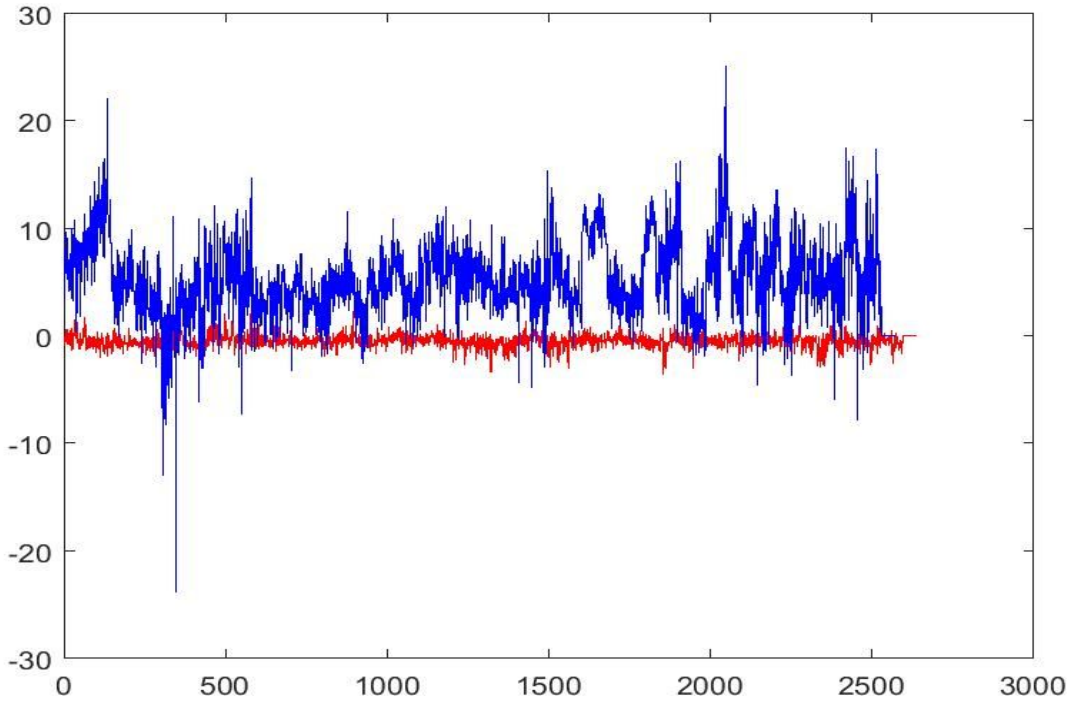
1. **Kurtosis (Principle Component 1)**



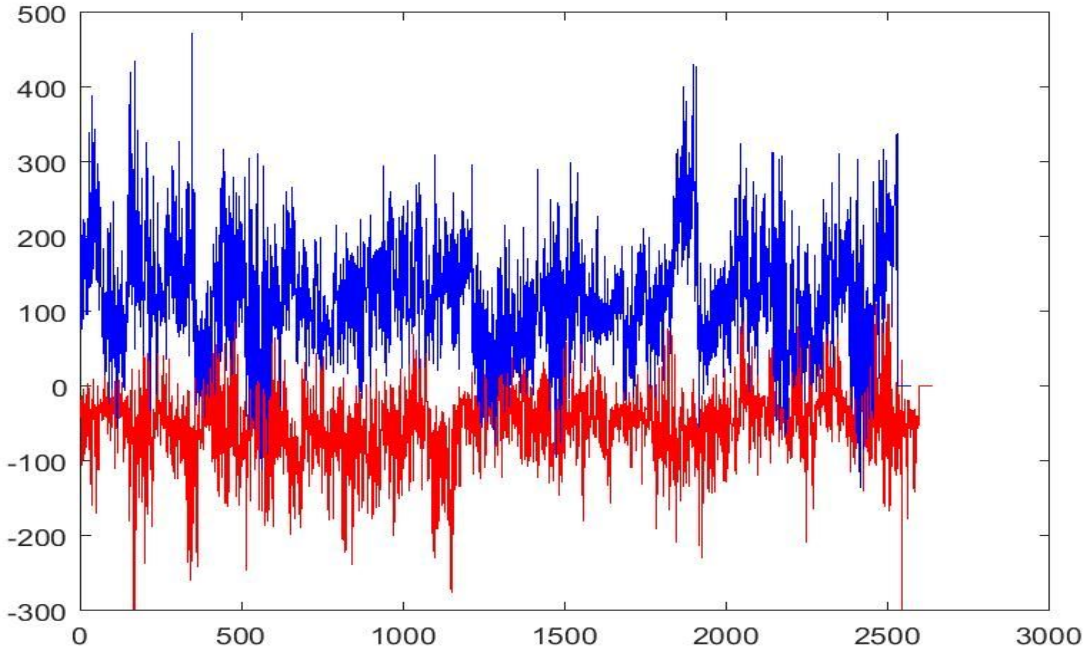
2. Kurtosis (Principle component 2)



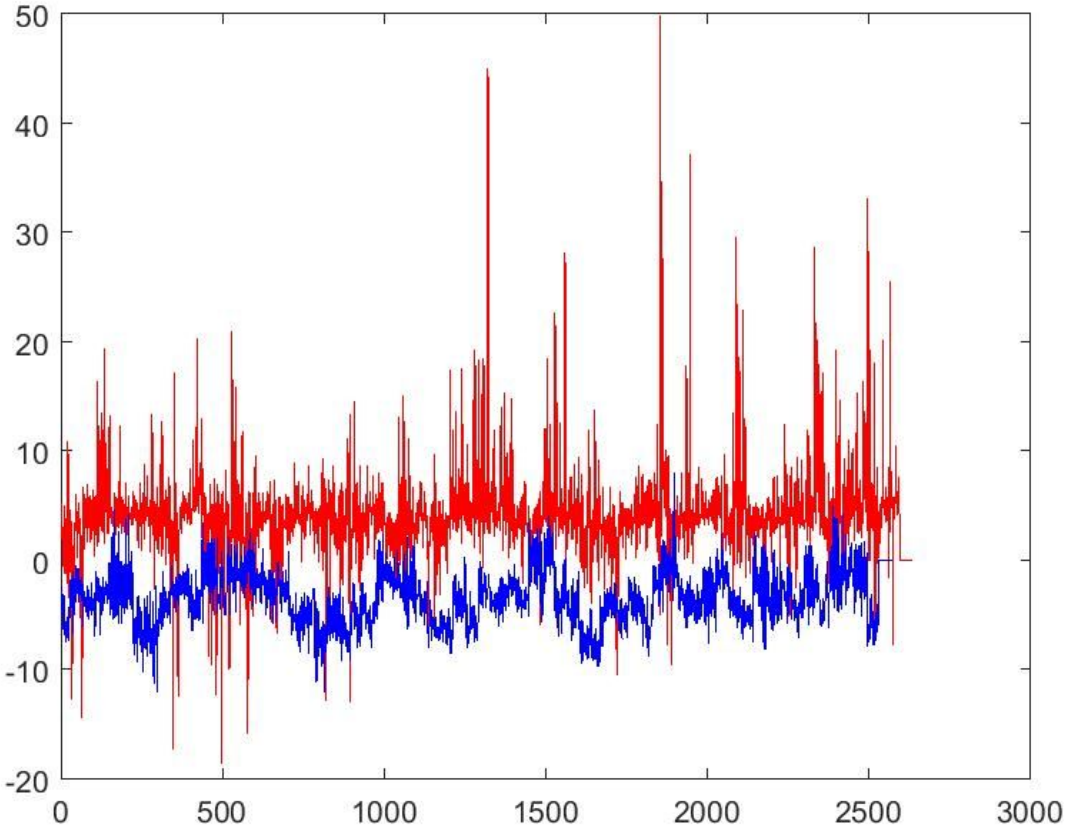
3. Kurtosis (Principle component 13)



4. Skewness (Principle component 2)



5. Variance (Principle component 14)



Subtask 5: Argue whether doing PCA was helpful or not. May be compare the plots generated from subtask d of task 2 and subtask 4 of Task 3.

If the features selected are good enough to differentiate between two classes that is eating and non eating, doing PCA would not make any sense. The features we have selected, more or less distinctly identify the two actions. We cannot compare the results of PCA and the results obtained from the task-2 because task-2 tells which sensor and the corresponding feature is best to distinguish between the two actions. Whereas, in PCA we lose the information about the sensor and the new features are the linear combination of the old data.

However, PCA tries its best to categorise the two actions and gives a distinction comparable to that of task-2 (in some cases). But we cannot say which ‘sensor’ is a good attribute to distinguish between actions after doing PCA. We can only say which ‘feature’ collectively is better.