CSE 572 Assignment#1
Ryan Meagher

PHASE 1:

The first challenge of this project was getting the root directory containing all of the different ground truth files and IMU/EMG files for each user's spoon/fork actions into a workable format. I choose to use jupyter notebook with python 3.5 as my workspace.

I ran my jupyter notebook from the root directory of the zip folder that was provided to us. There was a mismatch between the naming convention between user9 and user09, so steps were taken to handle this when stepping through the directory with pythons os module. In addition to this, user19_spoon had a space as a delimiter in row 4 instead of a comma, so in order to properly apply the *100/30 manipulation of start and end frames, I had to implement code to change row 4 to the correct formatting. One of the rows of another user also had an additional comma but pandas.read_csv function was able to work through this on its own.

I extracted the data for appropriate users from both the IMU and EMU data; however, there were timestamp discrepancies between many of the users IMU and EMG data. For example, user39_fork EMG data has 38241 rows while the IMU data has only 18562 rows. In addition to this, some of the start and stop rows from the users ground truth file are out of bounds in the corresponding IMU and EMG user data. If this was the case, I took the users data from the point up until where the ground truth went beyond the number of rows in IMU or EMG files as to not lose an entire users' data.

I sorted the data into two larger matrixes, an eating and a non-eating matrix. For the non-eating matrix, I randomly sampled the exact same amount of eating samples per user from all of their non-eating rows. In doing this, I achieved a class balance and the non-eating matrix was a representative sample of all users non-eating actions.

In addition to this I created a dictionary containing each individuals IMU non-eating and eating matrixes as well as their EMG non-eating and eating matrixes so that I could examine each individual's data separately. I wanted to do this so I could paint a better picture of the classification task. Maybe some eaters are much more difficult to classify comparatively to others and may even be outliers to other users with regards to how they perform their eating actions. If that were the case these eaters should be thrown out. This will be very useful when I start to implement a model and evaluate the performance on individual users as well as the entire cohort of users, this is when it will become clear whether certain individuals should be omitted or not.

PHASE 2:

The first thing I decided to do was to reset the indexes of both eating and non_eating matrixes and then take a plot of 'EMG8' vs the index of the matrixes of both the eating and non_eating samples to get a better look as to how the data was distributed and spread out across the feature space. As you can see from figure1 below, both eating and non_eating contain spikes in the data so maybe our start and stop frame need to be slightly ajusted but one cannot really say as there is no additional access to information regaurding the videos from

which the ground truth was captured from. However, given the EMG8 data in figure 1 we can start to infer about some good methods for feature extraction.

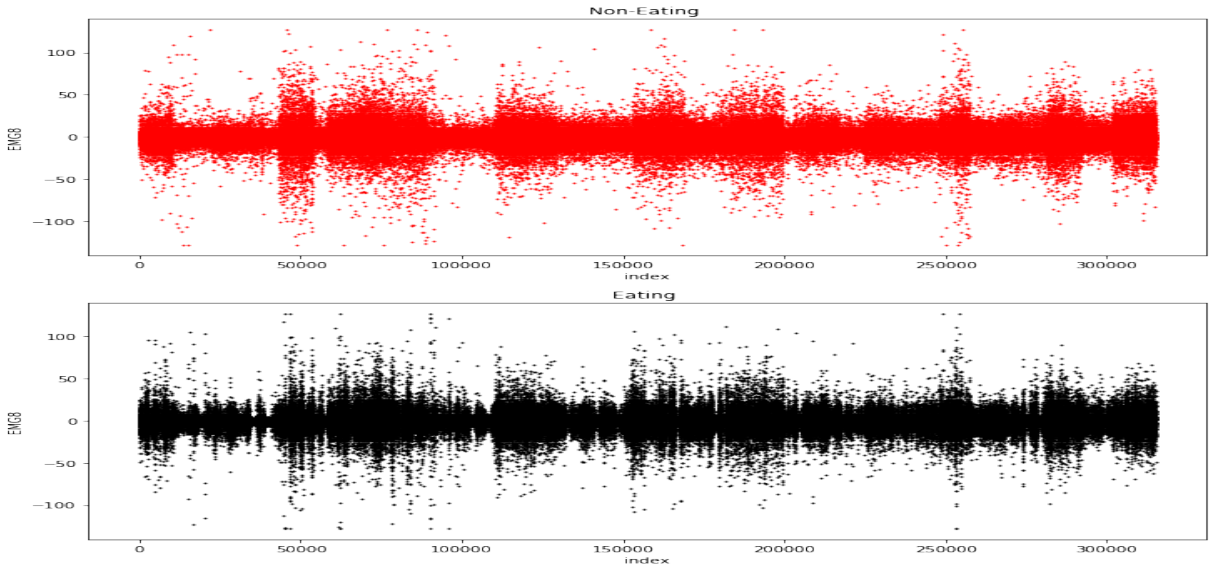


Figure1: EMG8 with no transformations is plotted for both eating and non-eating actions.

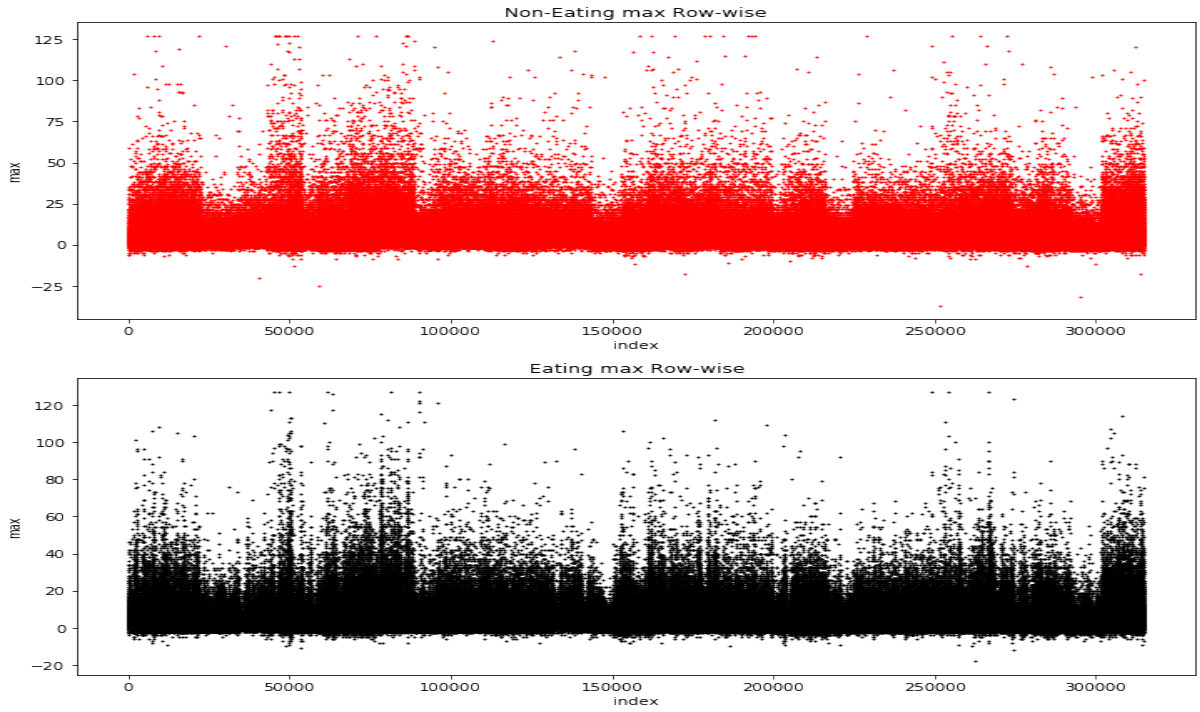
Just from looking at figure 1 we can see that there is a much greater spread in shorter periods of time in eating actions of EMG 8 comparatively to the non-eating actions. Given this spread I believe that variance and standard deviation of the EMG1-EMG8 sensors calculated for every row would be a good feature extraction method. In addition to this, the drastic short burst spikes lead me to believe that taking the minimum, maximum and range would also be good features to extract; however, I am a little hesitant of these features as the non-eating data also exhibits maximums and minimums that are comparable to eating actions. Furthermore, from the large positive and negative spikes, I thought that maybe taking the absolute sum across every sensor would be a good feature extraction method as well as the, magnitude/Euclidian norm of the row vector, and finally the root mean square of a row vector along with the mean. I thought that due to these large quick spikes over the 8 sensors, the magnitude, mean and RMS of the eating data would be good features for getting some separation between eating and non-eating classes. That leaves us with a total of 9 statistical features to extract: mean, root mean square (rms), vector magnitude, absolute sum of all values, min, max, range, standard deviation and variance.

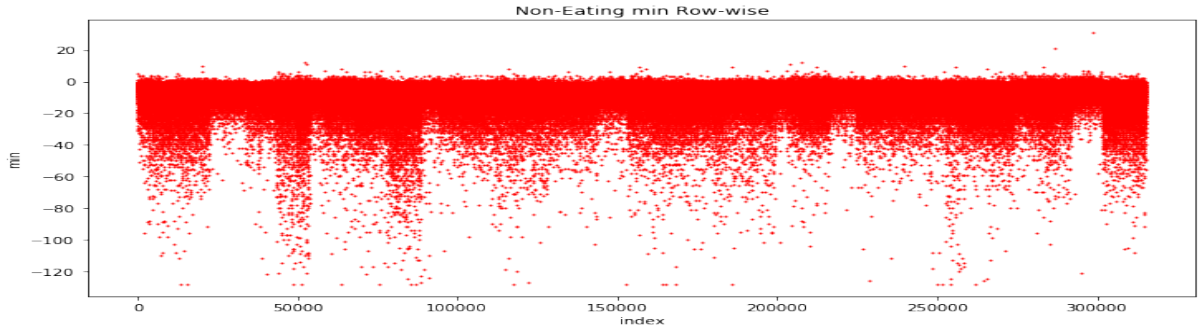
Before I did any feature extraction, I personally believed that although there is a lot of overlap between the 2 class features, the greatest distinction between classes would come from the RMS, magnitude, standard deviation and variance of the row vectors.

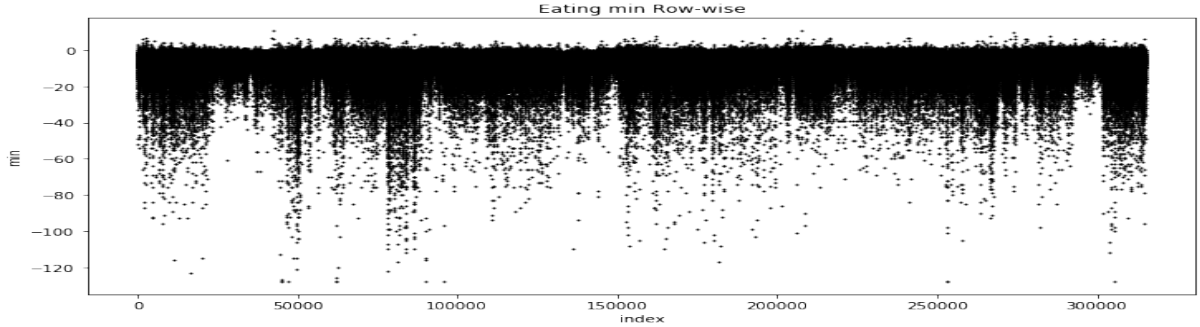
To apply this, it was relatively straightforward using the panda's module. All that needed to be done for mean, standard deviation, variance, min, max was to use the appropriate pd.DataFrame function along axis=1 for applying these functions across the rows. The only

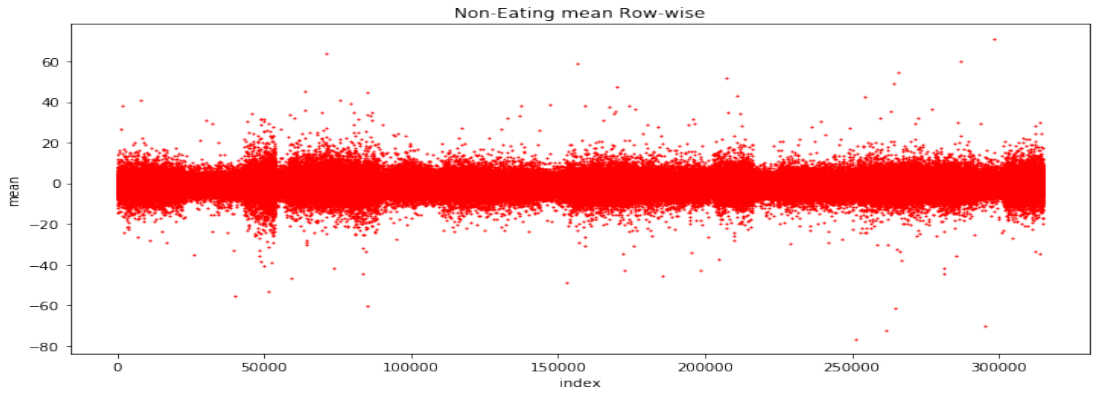
ones that were slightly more difficult to apply were the absolute sum of all the values, root mean square, range and the magnitude. For the absolute sum of values, I just needed to take the absolute value of the whole data frame and then get the sum of every row. For the Euclidian norm which is very similar to the RMS I took the squares of every value in the row and then square rooted these summed squared vales. To apply RMS I made a slight modification to this where instead of taking the sum of the squares I took the mean of the squares and then I square rooted that value. Finally, I took the range by subtracting the maximum value of the row with the minimum value of the row.

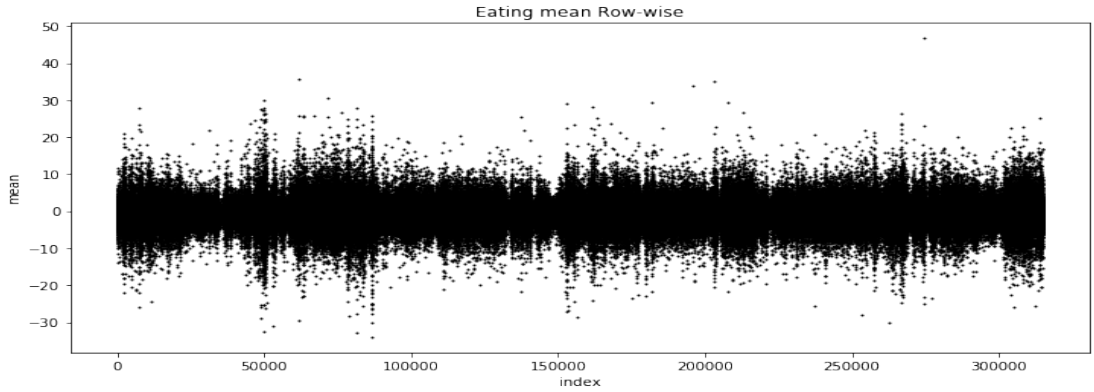
Below are the plots obtained from applying the aforementioned statistical analyses row-wise on all users EMG data.

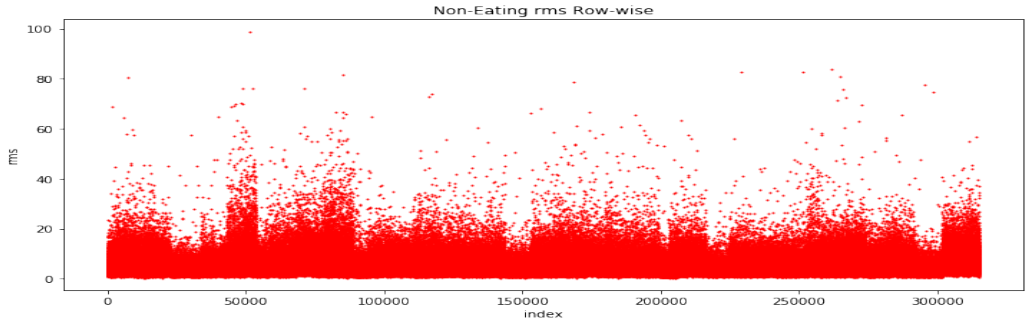


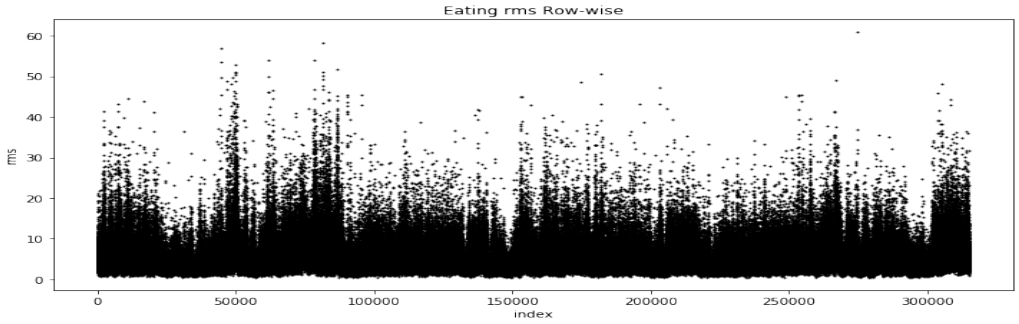

Non-Eating min Row-wise
min
index

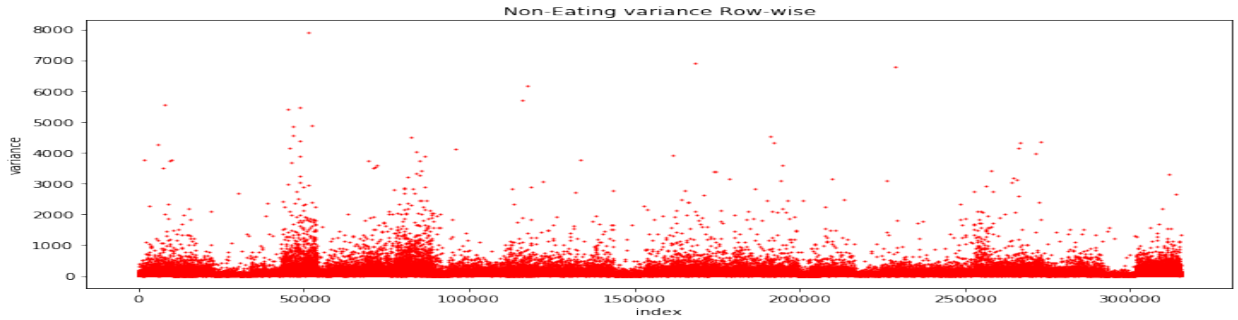

Eating min Row-wise
min
index

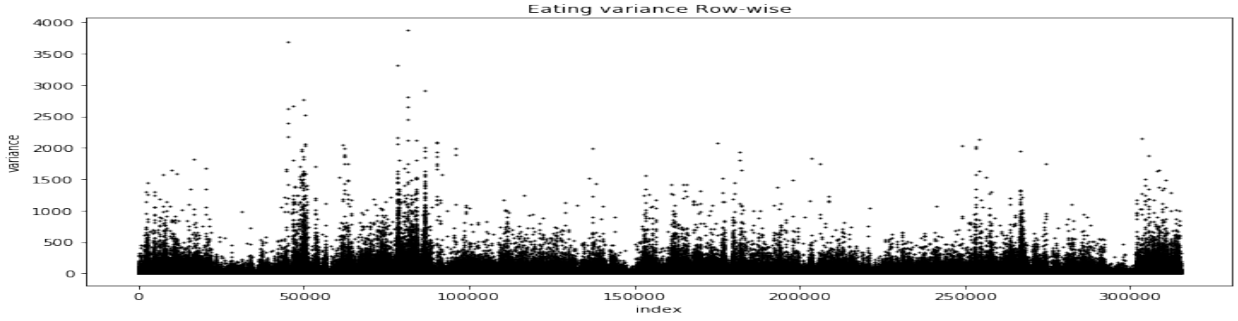

Non-Eating mean Row-wise
mean
index

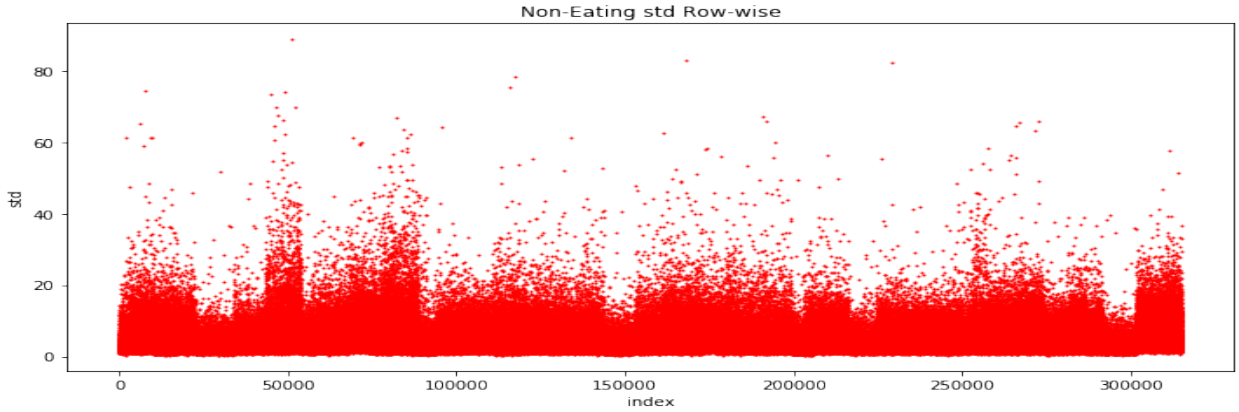

Eating mean Row-wise
mean
index

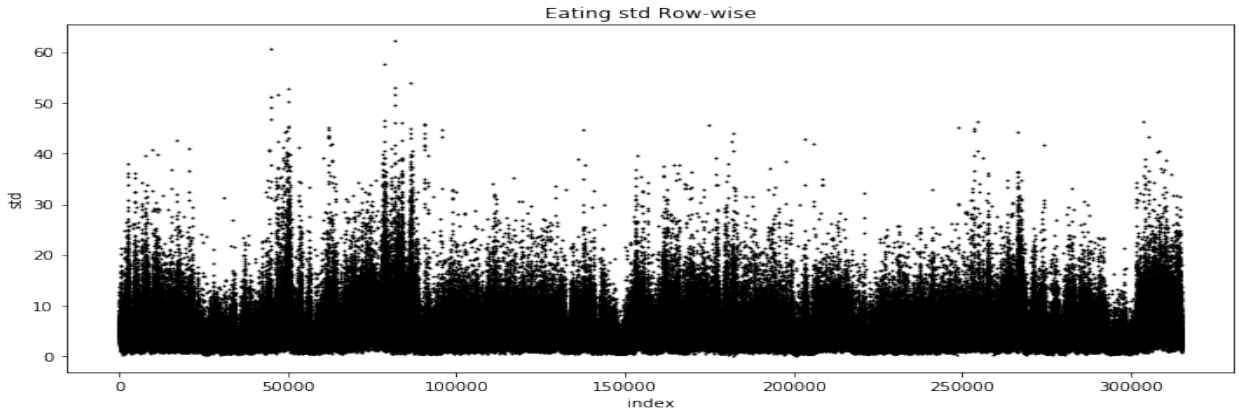

Non-Eating rms Row-wise
rms
index
0
20
40
60
80
100
0
50000
100000
150000
200000
250000
300000

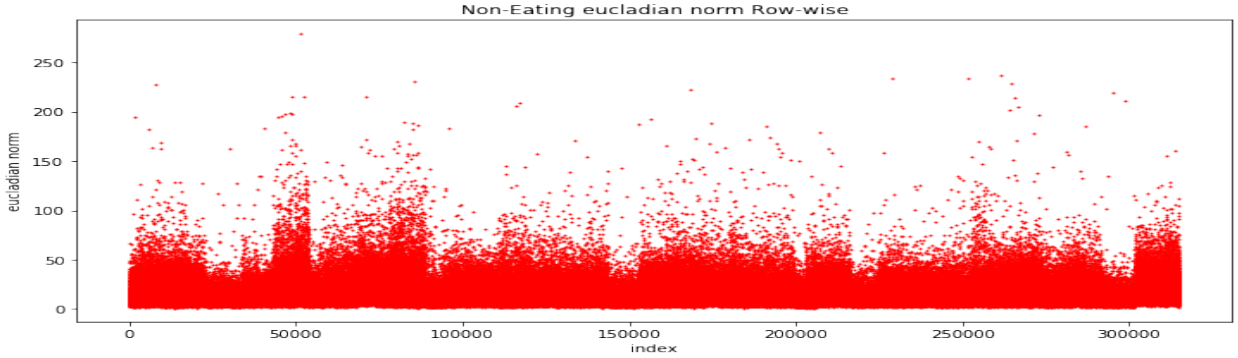

Eating rms Row-wise
rms
index
0
10
20
30
40
50
60
0
50000
100000
150000
200000
250000
300000

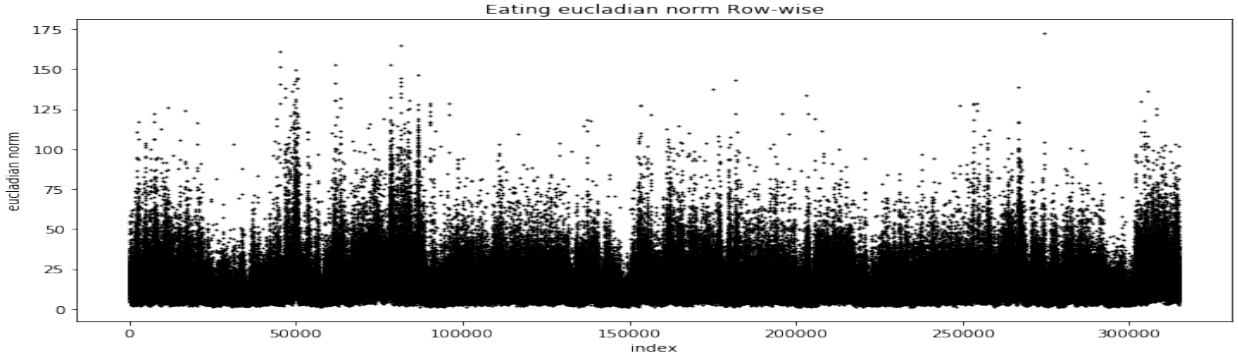

Non-Eating variance Row-wise
variance
index
0
1000
2000
3000
4000
5000
6000
7000
8000
0
50000
100000
150000
200000
250000
300000

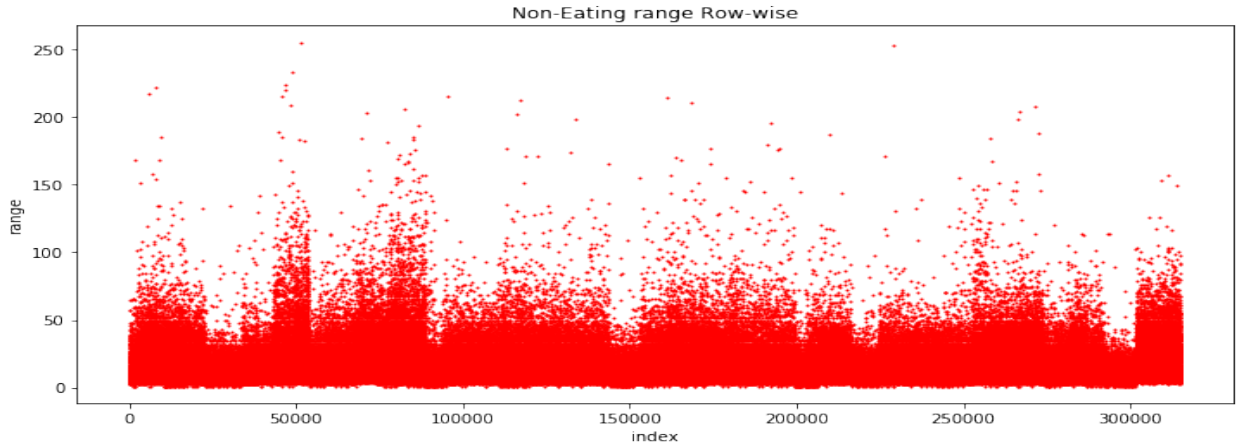

Eating variance Row-wise
variance
index
0
500
1000
1500
2000
2500
3000
3500
4000
0
50000
100000
150000
200000
250000
300000

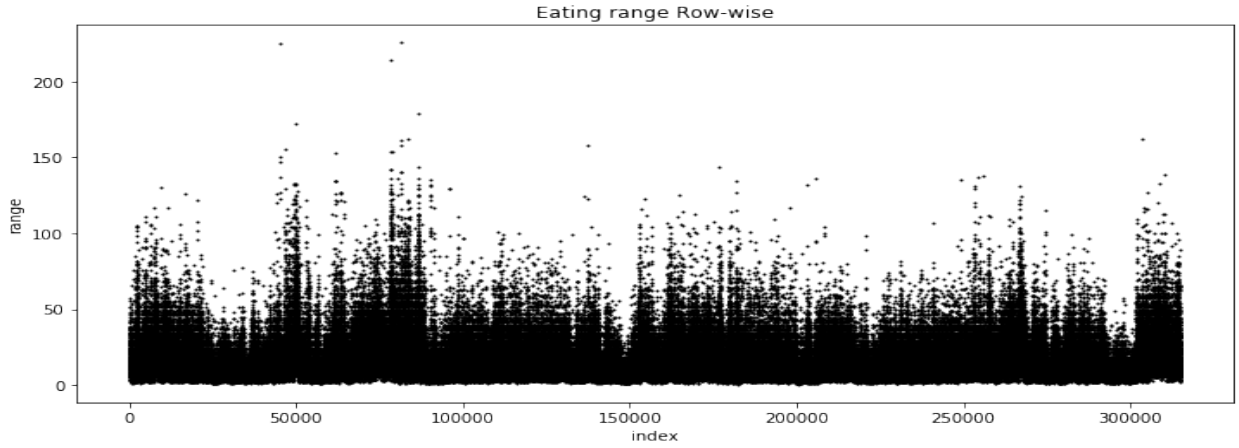

Non-Eating std Row-wise
std
index
0
20
40
60
80
0
50000
100000
150000
200000
250000
300000

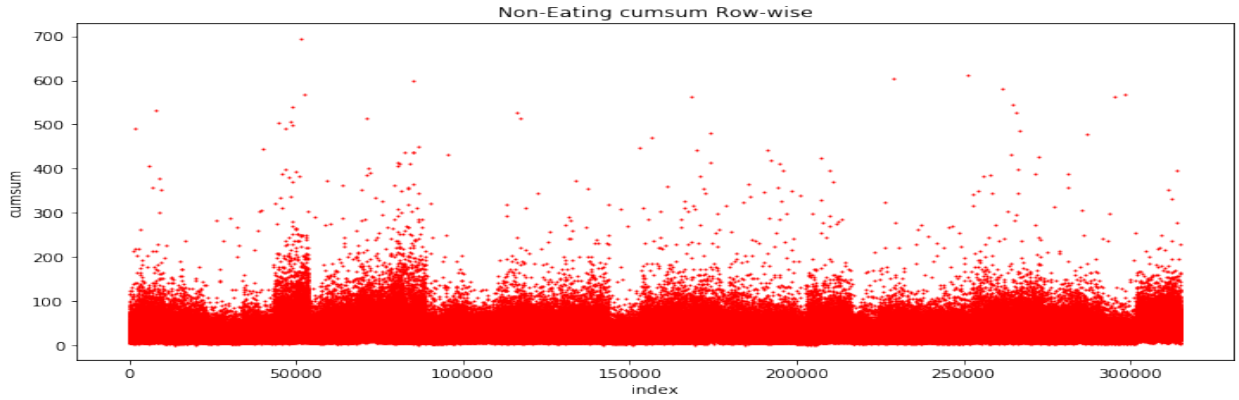

Eating std Row-wise
std
index
0
10
20
30
40
50
60
0
50000
100000
150000
200000
250000
300000

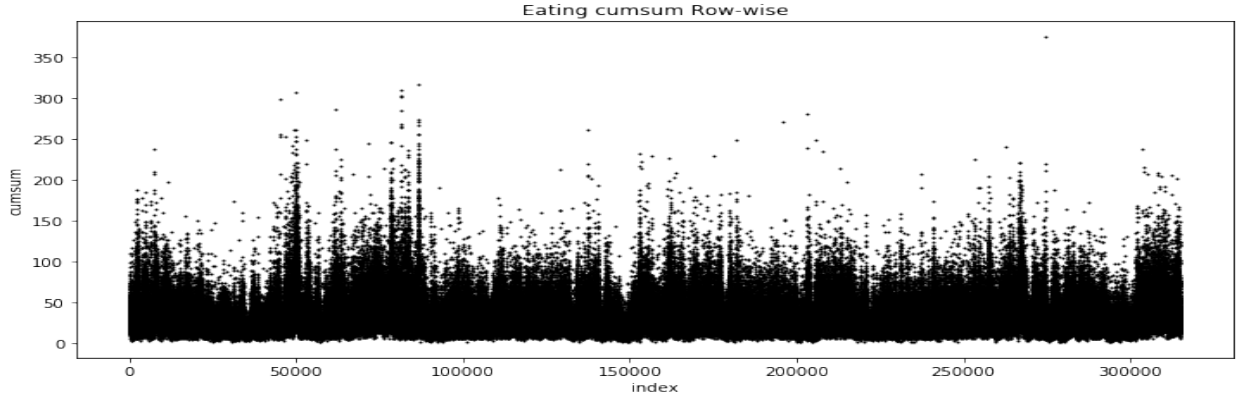

Non-Eating eucladian norm Row-wise
eucladian norm
index
0
50
100
150
200
250
0
50000
100000
150000
200000
250000
300000


Eating eucladian norm Row-wise
eucladian norm
index
0
25
50
75
100
125
150
175
0
50000
100000
150000
200000
250000
300000


Non-Eating range Row-wise
range
index
0
50000
100000
150000
200000
250000
300000
0
50
100
150
200
250

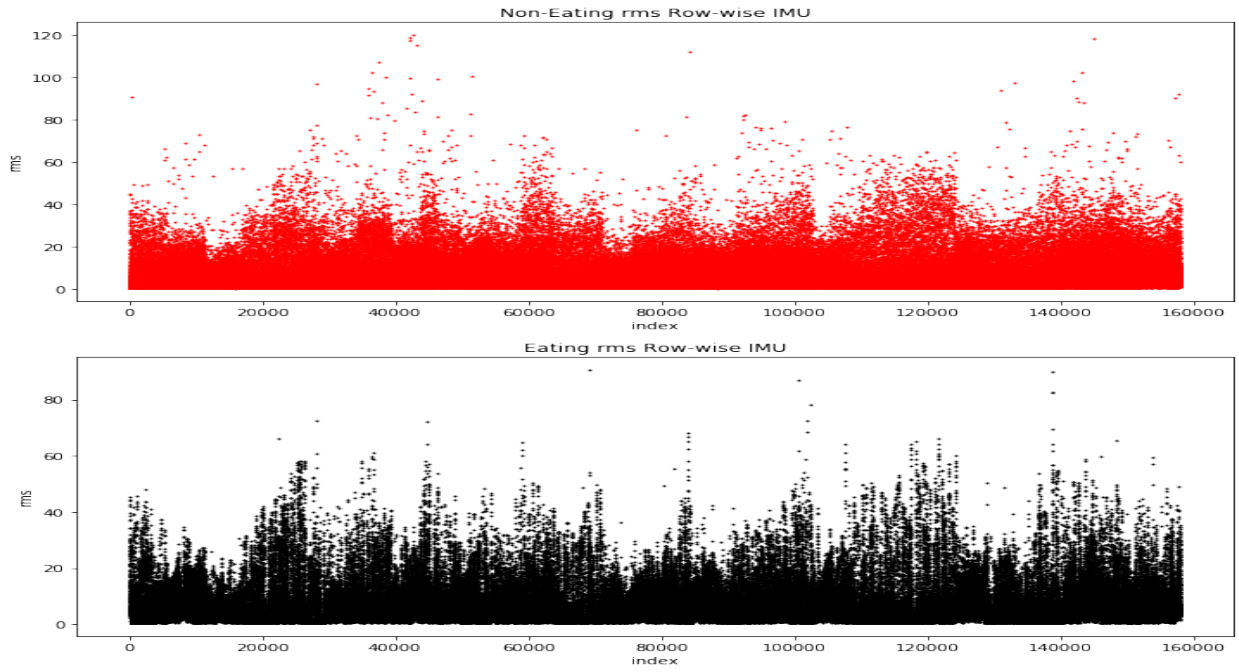

Eating range Row-wise
range
index
0
50000
100000
150000
200000
250000
300000
0
50
100
150
200

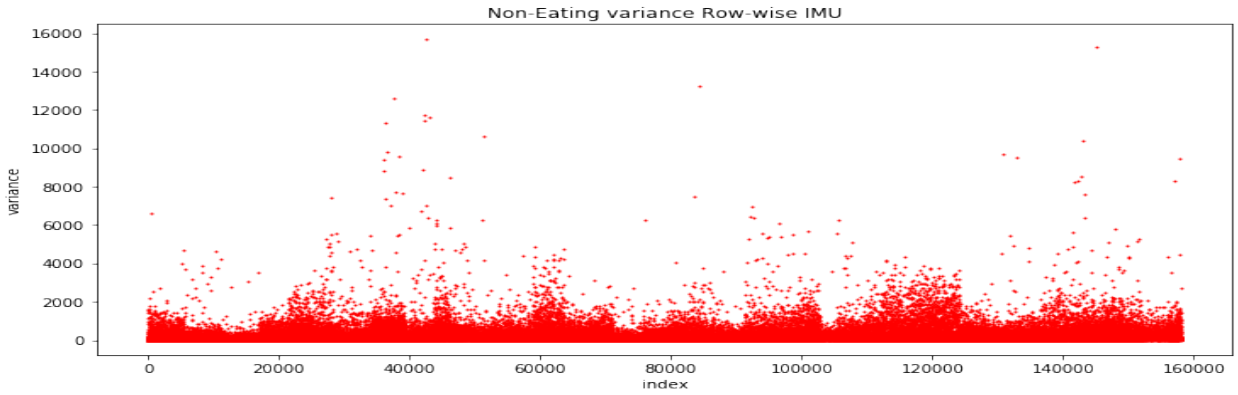

Non-Eating cumsum Row-wise
cumsum
index
0
50000
100000
150000
200000
250000
300000
0
100
200
300
400
500
600
700

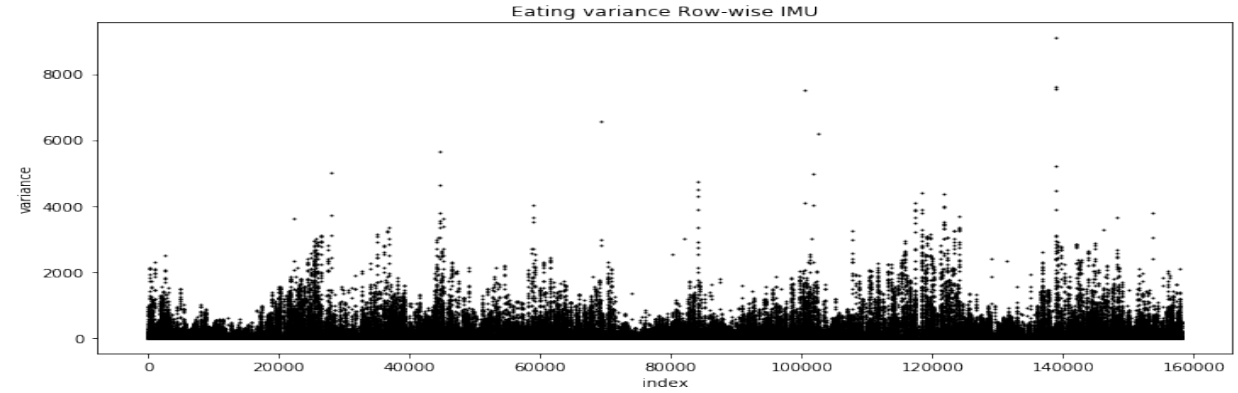

Eating cumsum Row-wise
cumsum
index
0
50000
100000
150000
200000
250000
300000
0
50
100
150
200
250
300
350

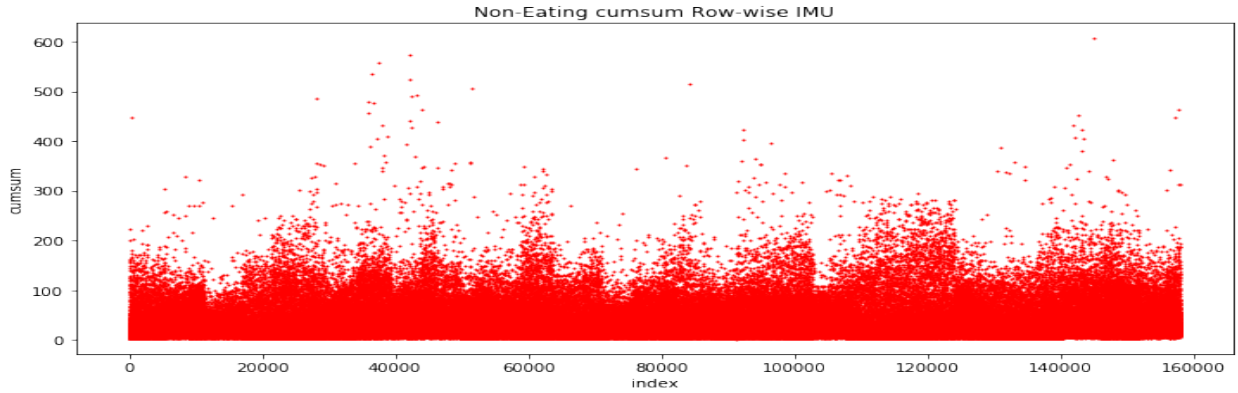
I think that my initial intuition holds pretty well after looking at these 9 statistical features. I think that min, max and the range of the rows show the least amount of separation which was what I was kind of assuming as they both had groups of points that had pretty positive and negative numbers. I think that the standard deviation, variance, sum of the absolute values, and the RMS show the greatest separation between the 2 classes. I especially like the sum of the absolute values, the Euclidian norm/magnitude and the standard deviation. I feel like those 3 statistical features give the greatest separation between classes and that was kind of what I was expecting after seeing how the eating actions consisted of sharply spiked data.

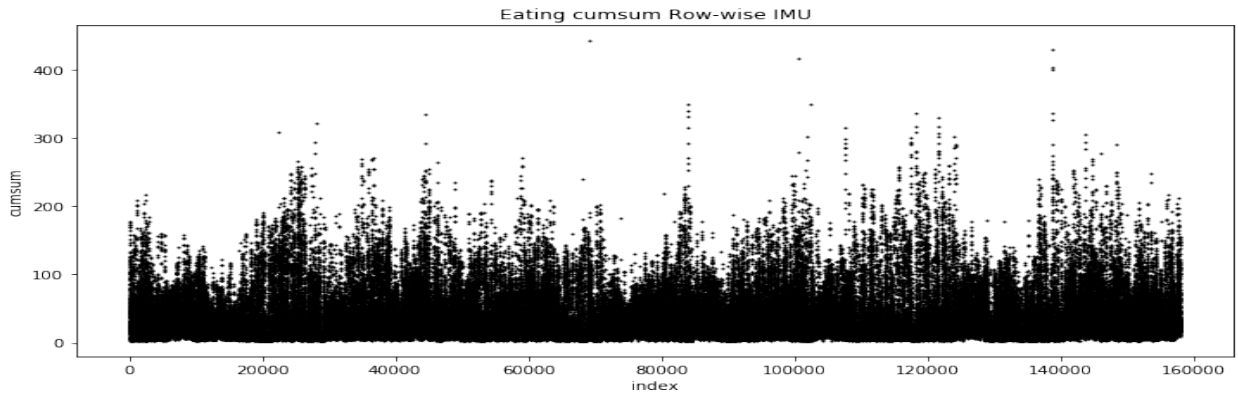
Just to compare what the top features of the EMG data with the IMU data I have provided a few graphs below of IMU features of the RMS, sum of absolute values, variance and eucladian norm and these features also seem to show pretty good separation of classes in the IMU data.

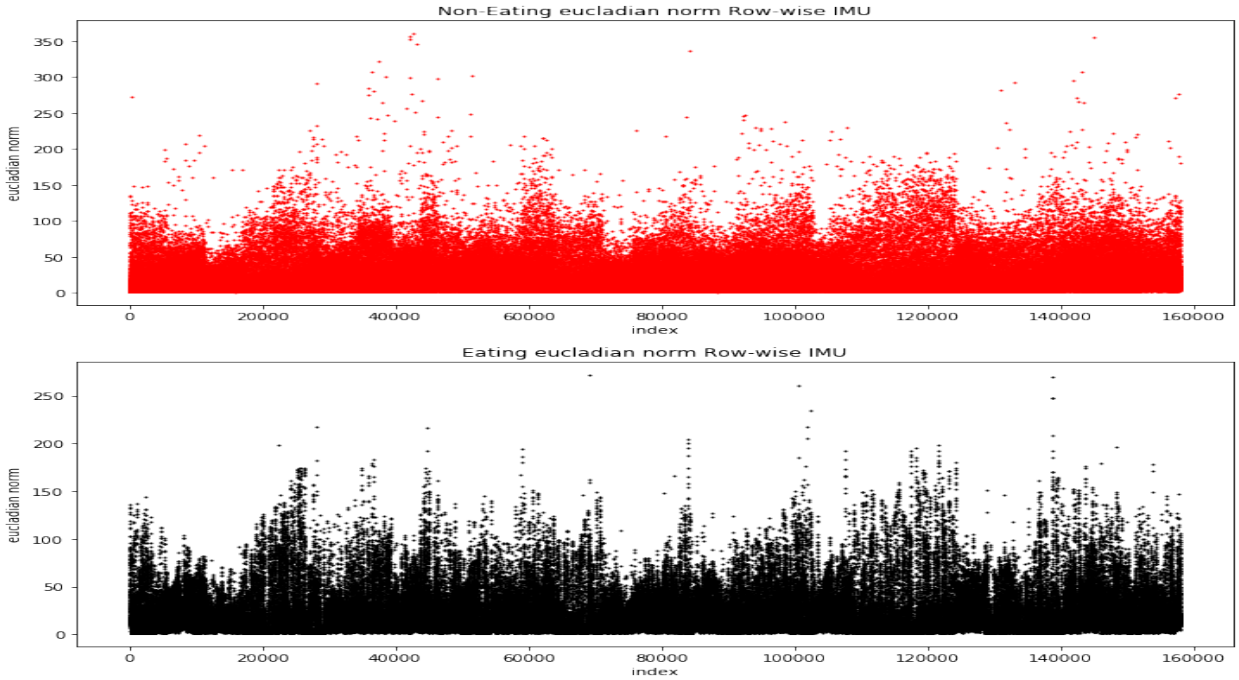



Non-Eating variance Row-wise IMU
variance
index
0
2000
4000
6000
8000
10000
12000
14000
16000
0
20000
40000
60000
80000
100000
120000
140000
160000


Eating variance Row-wise IMU
variance
index
0
2000
4000
6000
8000
0
20000
40000
60000
80000
100000
120000
140000
160000


Non-Eating cumsum Row-wise IMU
cumsum
index
0
100
200
300
400
500
600
0
20000
40000
60000
80000
100000
120000
140000
160000


Eating cumsum Row-wise IMU
cumsum
index
0
100
200
300
400
0
20000
40000
60000
80000
100000
120000
140000
160000



PHASE 3:

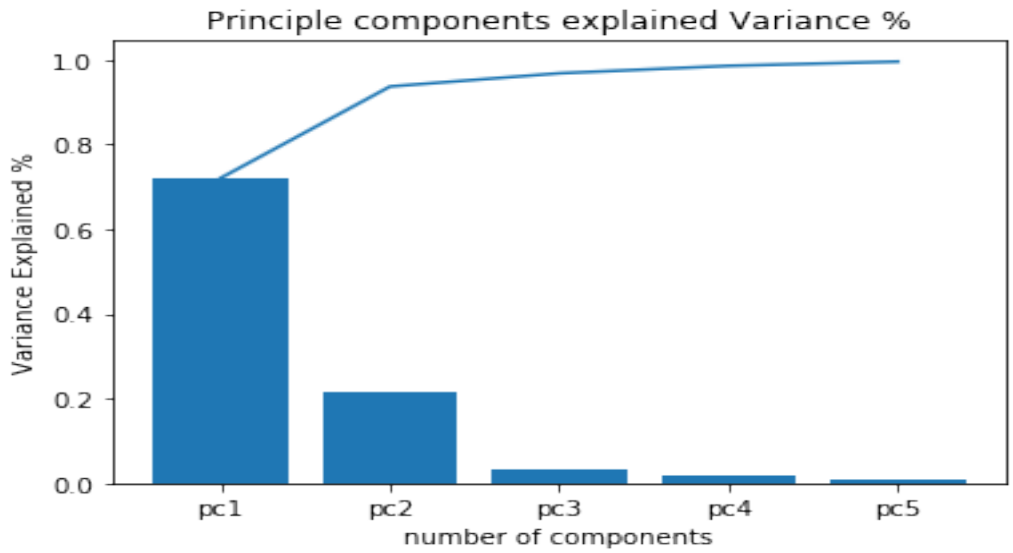
PCA will only be applied to EMG for the sake of clarity in this paper. In order to do this, we need to combine our eating matrix that has the statistical features that have been extracted row-wise with our non-eating matrix with the same statistical features and the same data frame structure. To make our pca matrix we will concatenate the non-eating feature matrix to the bottom of the eating feature matrix. We then need to drop the class column which will be used later. It is easy to infer what rows correspond to which class in the pca matrix as the 1st half of the data frames rows are class eating and the 2nd half are class not-eating.

In addition to this since all of our features that we extracted are on a different scale we either need to normalize the data or standardize the data. We will standardize the data by taking a column and subtracting every value in the column by the mean and then dividing by the standard deviation. This is done because since our attributes are on different scales, we don't want attributes with larger magnitudes contributing to most of the variance in pca.

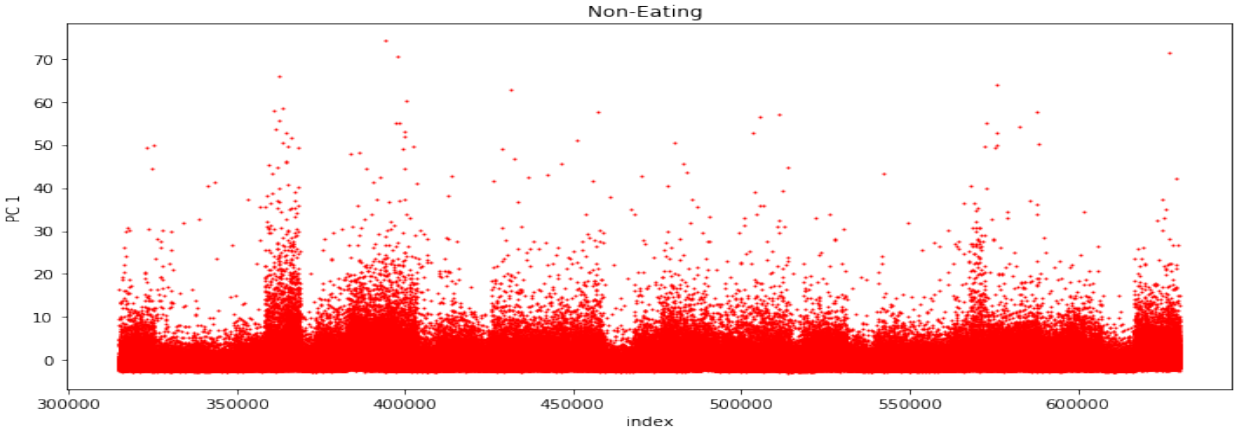
After standardizing the matrix, the covariance matrix is calculated from the scaled data via numpy.cov this generates a square matrix which allows us to apply the scipys liner algebra eig function to obtain a matrix [V,D]. The columns of V present eigenvectors of the covariance matrix and the diagonal of matrix D contains the eigenvalues which is the magnitude of the eigenvector and has the same index as eigenvalue, signifying it's the amount of variance it contributes from the scaled dataset. Now that we have our eigenvectors, we can find the feature data by multiplying the scaled data by the inverse of these eigenvectors. Now we can use the eigenvalues (magnitude of the variance) to sort the feature data in descending order to principle component 1,2.. etc based on the number of components that we want or until a

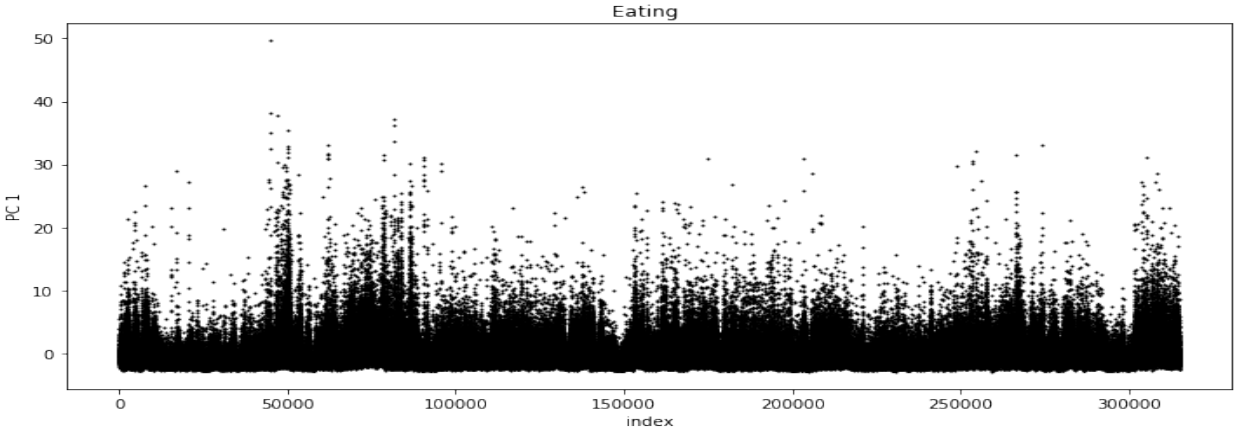
certain variance contribution has been achieved. Instead of taking all these steps we can use sklearn.decomposition.pca which calculates this new feature matrix for us as well as tells us how much variance each principle component contributes.

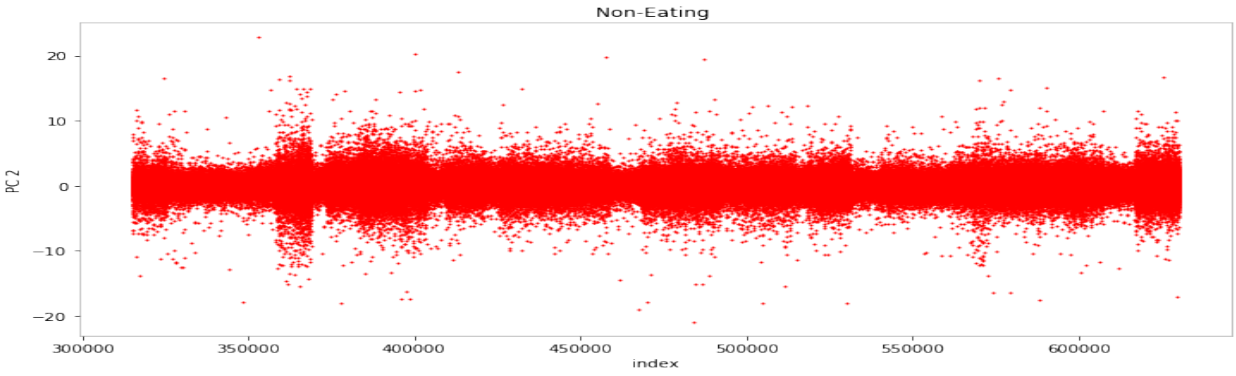
After applying PCA and obtaining the new matrix I found that my first 2 principle components contribute 93.7% of the explained variance as can be seen in my figure below.

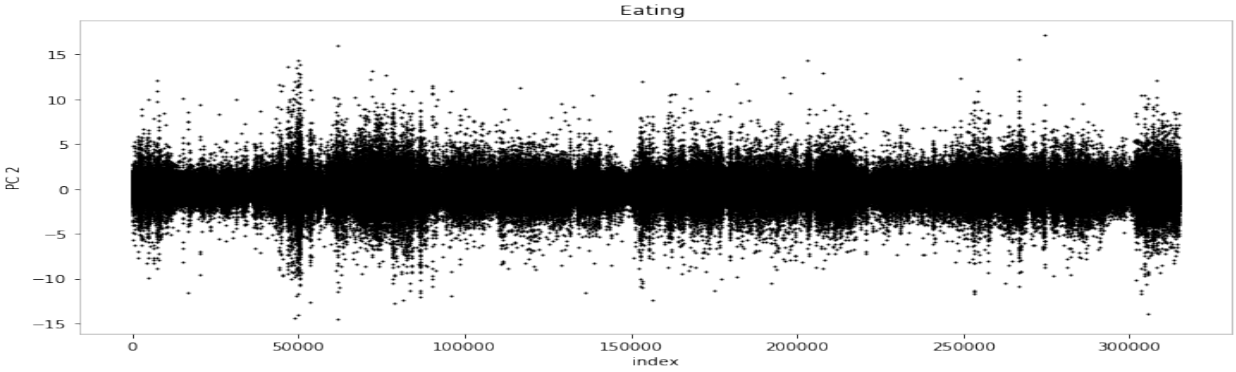


I decided to split my pca matrix in half keeping the same order of eating vs non-eating as their index is directly correlated to the users which the data is taken from. One users index for eating actions will be directly applicable to the non-eating index as the same number of samples from a user was added to eating and then non-eating respectively and the process was repeated.
I only plotted Principle component 1 and principle component 2 as they are the components which contribute the most to the variance. Principle component 1 seems to have some decent separation between the classes, but I am not sure it is any better than some of my feature extraction methods such as standard deviation, sum of the absolute values, or the magnitude of the vectors. Principle component 2 on the other hand doesn't have nearly as good of separation and is comparable to that of min, max, range and it looks incredibly similar to the mean. I'm thinking that principle component 2 captured a lot of the variance seen in those features which didn't show as good of separation in phase 2 while Principle component 1 was able to capture more of the variance from the features in phase 2 which showed a better distinction between the classes. It will be hard to tell which features will give me better performance for classification until I actually start implementing some of the models. I really think that the absolute sum of EMG1-EMG8 is going to be a good indication between classes because it looks to have the best separation. Still though, I don't have nearly as much separation as I would like between my classes and it will be interesting to see how different combinations of my current data will perform.


Non-Eating
PC 1
index


Eating
PC 1
index


Non-Eating
PC 2
index


Eating
PC 2
index