Activity Recognition Project

CSE 572 – Data Mining

Group Number: 16

Partners

- Abhyudaya Srinet - 1215201260
- Darshan Dagly – 1215180174
- Pranay Jagtap - 1215139991
- Saurabh Abhale – 1213185270
- Shreya Darak - 1215202846

Phase 1: Data collection

Data collection for this project was achieved using a Myo Armband.
Activities recorded during Phase 1:

1. Eating
2. Cooking

Phase 2: Feature Extraction

We have chosen the activities Eating and Cooking to perform our analysis on.

Since the sampling frequency of the dataset is not even, we perform a linear interpolation of the respective sensor data to get an even sampling frequency. This interpolation is performed for every chosen feature.

Following methods were selected and implemented for feature extraction:

1. Standard deviation
2. Root Mean Square
3. Mean
4. Max
5. Fast Fourier Transform

We have used python libraries like numpy, pandas and sklearn to perform manipulations on the data. The data is loaded into matrices(DataFrame) using the pandas library.

Feature Extraction Methods:

1. **Standard Deviation**
   Standard deviation tells us how far data points are spread from the mean value. The inbuilt function in python for the dataframe matrices are used to calculate the standard deviation values.

2. **Mean**
   Mean is the average score calculated for a variable. The inbuilt function in python for the dataframe matrices are used to calculate the mean values.

$$X\_mean = (\Sigma x_i) / n$$

3. **Max**
   Max is the maximum value of a data point calculated over a data set. Max values of a human hand acceleration were calculated to represent each of the eating and cooking activities. The inbuilt function in python for the dataframe matrices are used to calculate the mean values.

4. **Root Mean Square**
   The Root Mean Square is calculated using the following formula:

$$x_{\mathrm{rms}} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \cdots + x_n^2\right)}$$

5. Discrete Fourier Transform

We use the python numpy library to calculate the DFT using the fast Fourier transform algorithm. These features were extracted from each of the time series and plotted onto a graph along with the raw data of each activity and an observation was made between all pairs of the 2 activities (EatFood1, EatFood2, EatFood3, EatFood4, Cooking1, Cooking2) to identify potential features.

List of features extracted

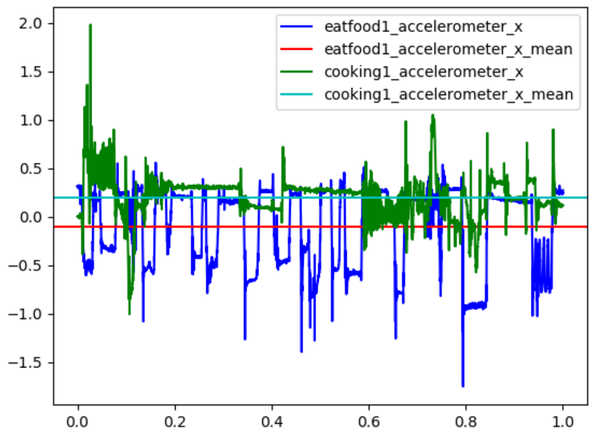
1. Mean of Accelerometer along X-axis:

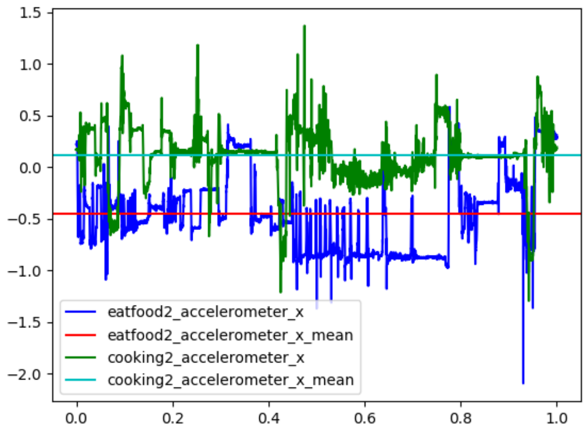
Intuition:

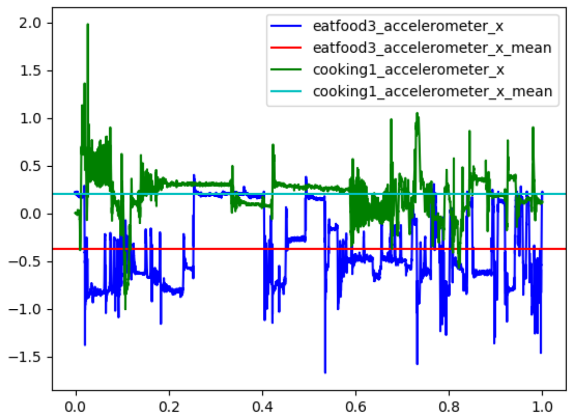
While eating or cooking food, acceleration of a human hand changes predominantly in the horizontal direction. Hence, a mean value calculated along the X-axis can be used to represent the corresponding action. The human hand tends to possess higher acceleration while cooking as compared to eating in the horizontal direction.

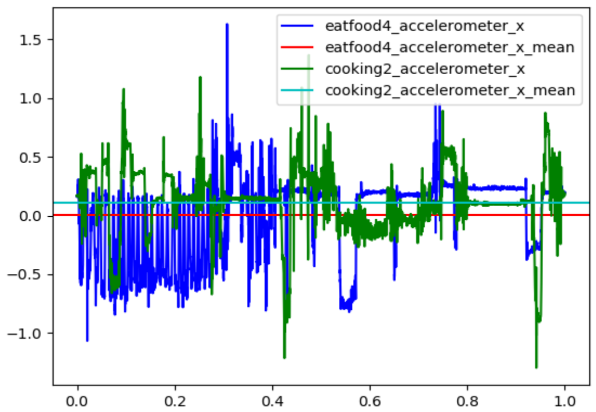
Observation:

The following plots compare the mean of accelerometer along the x-axis between the cooking and eating activity. It can be observed that the mean value of accelerometer along x-axis is always greater for cooking. **Hence, the intuition holds true**.









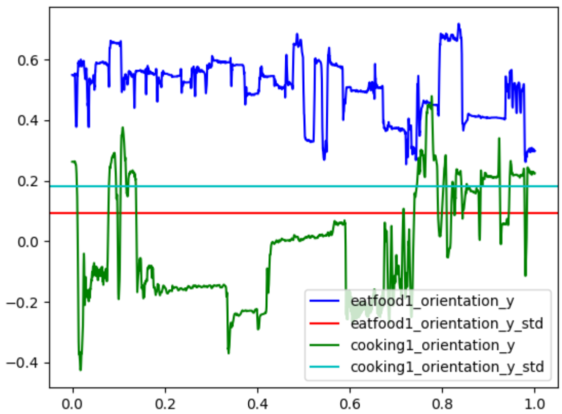
2. Standard Deviation of Orientation along Y-axis:

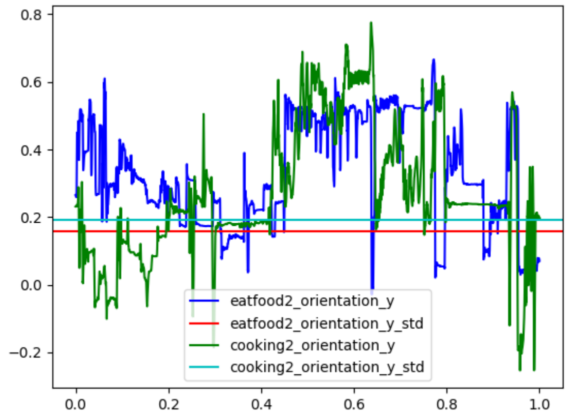
Intuition:

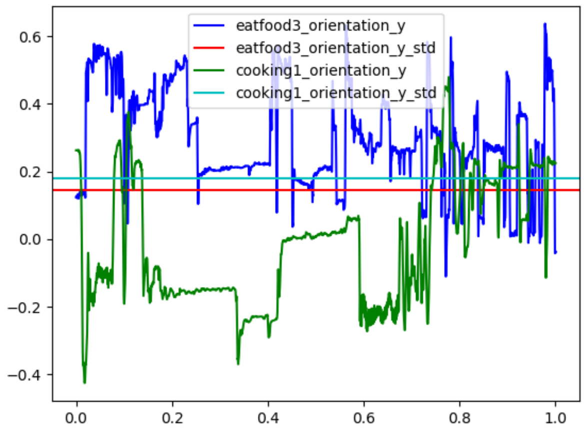
Orientation sensor will give us details of how the hand is tilted/rotated upside down or in other angles along 3 reference axes. Since cooking involves more tasks that require wrist movements like tilting, it will give us a better spread about its mean as compared to eating where movement in terms of rotation along these axes is relatively less.

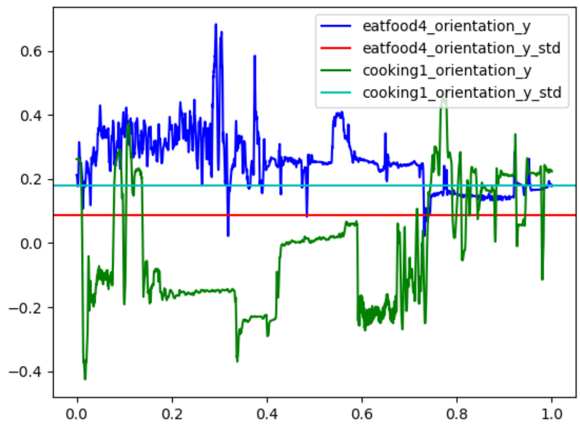
Observation:

The following plots indicate the standard deviation values for the orientation along y-axis along with the raw data. It can be observed that the standard deviation of the cooking activity is always greater than eating. **Hence, the intuition holds true.**









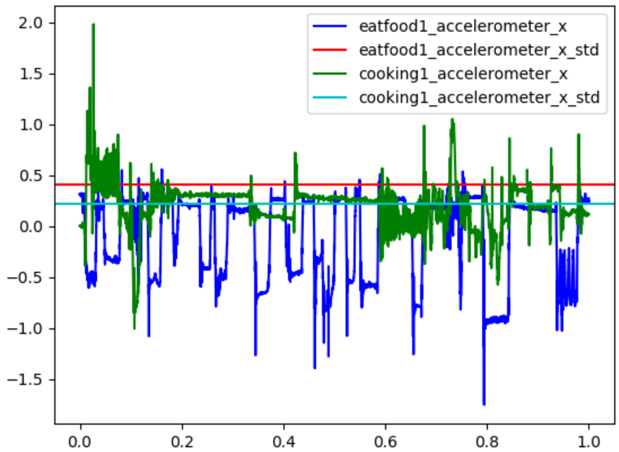
3. Standard Deviation of Accelerometer along X-axis:

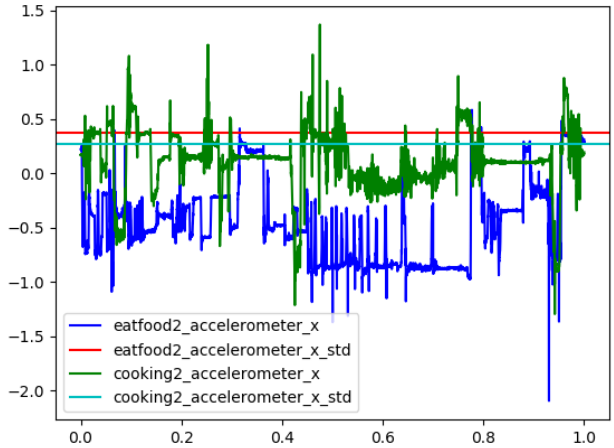
Intuition:

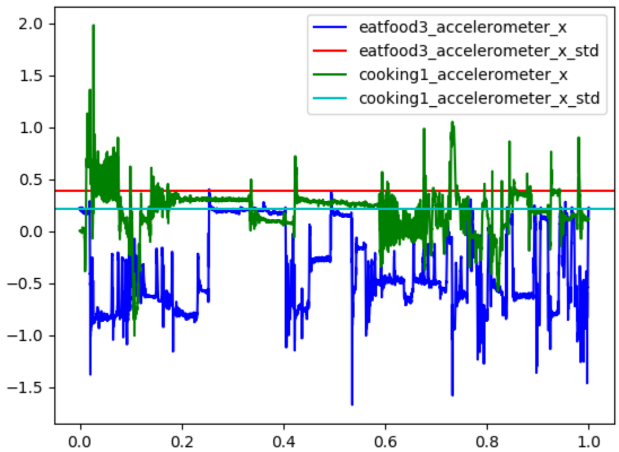
Accelerometer values must have the different range for different activities and therefore might be a useful method to distinguish between two activities. Since acceleration attributes will have higher values for cooking and comparatively lower values for eating, eating will have a smaller deviation from mean.

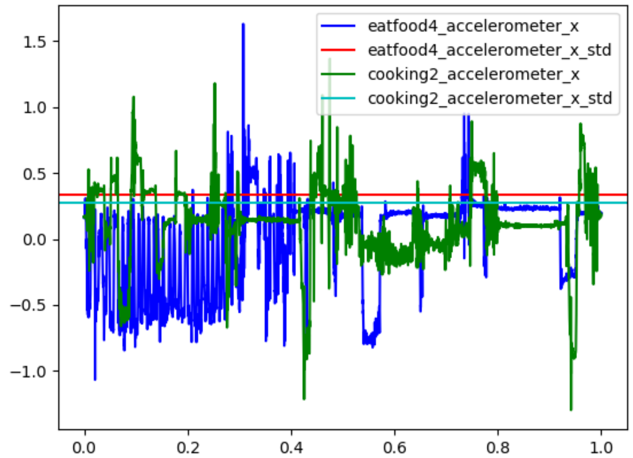
Observation:

The following plots indicate the difference of standard deviation values of accelerometer along x-axis between the eating and cooking activities. It can be observed that the standard deviation for eating is always higher than the standard deviation for cooking. **Hence, the intuition does not hold true.**









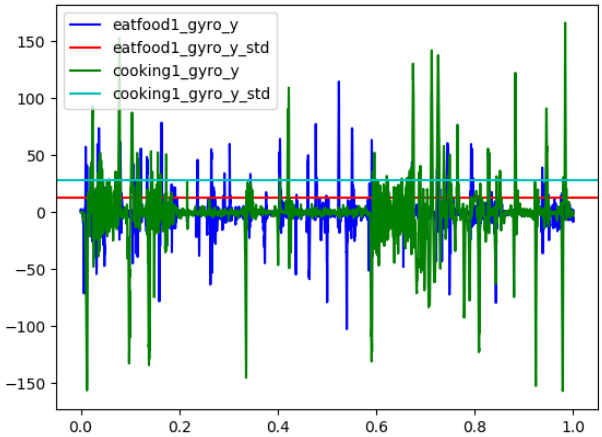
4. Standard Deviation of Gyroscope on Y-axis:

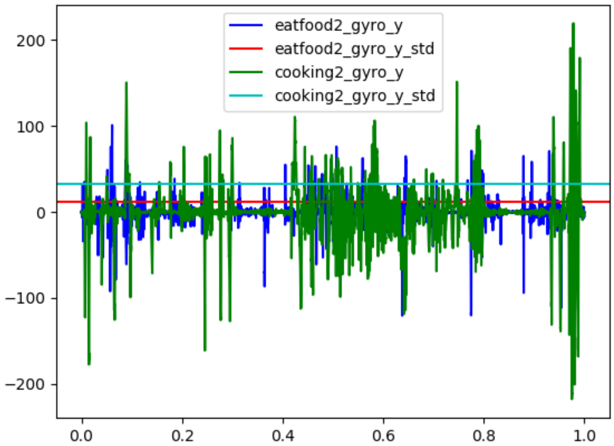
Intuition:

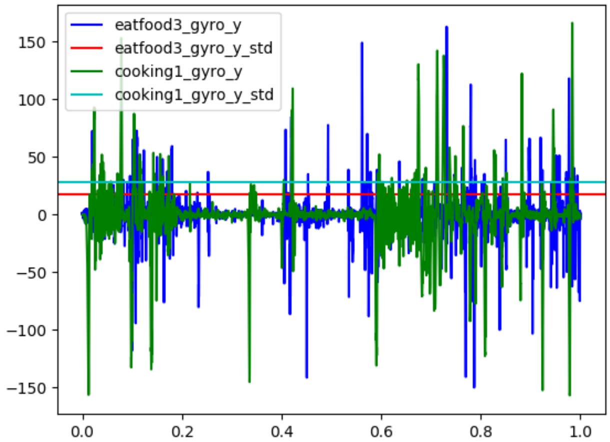
Gyro attributes give details of rotation and orientation of the hand. Since eating will have lesser spread as compared to cooking, finding values for its standard deviation should allow us to get clear distinction between these two tasks.

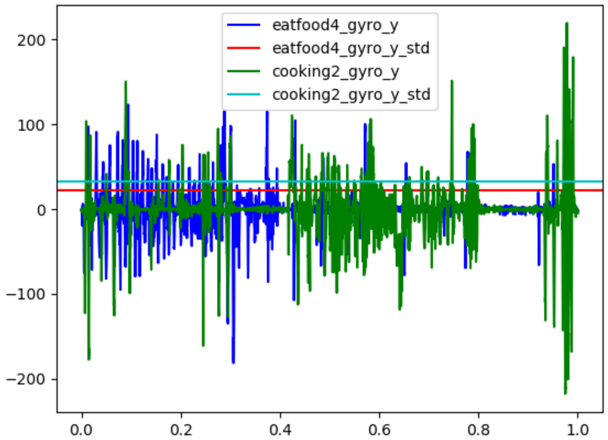
Observation:

The following plots indicate the difference between standard deviation of the gyroscope along y axis for eating and cooking activity. It can be observed that the standard deviation for cooking is always higher than the standard deviation for eating. **Hence, the intuition holds true.**









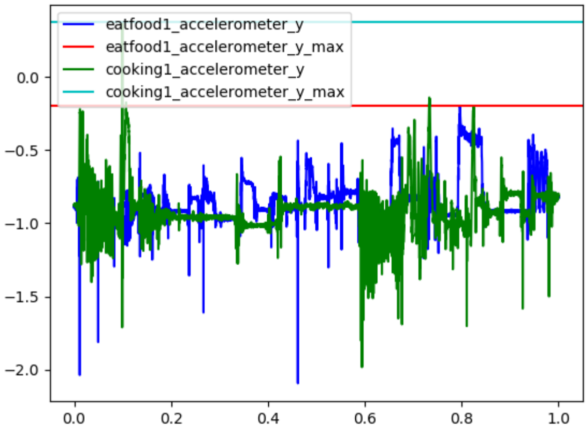
5. Maximum of Acceleration on Y-axis:

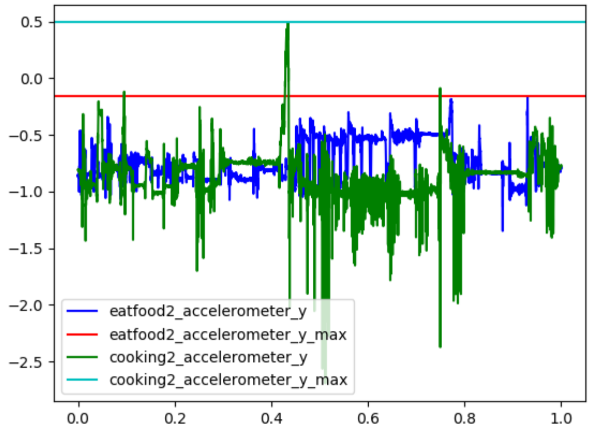
Intuition:

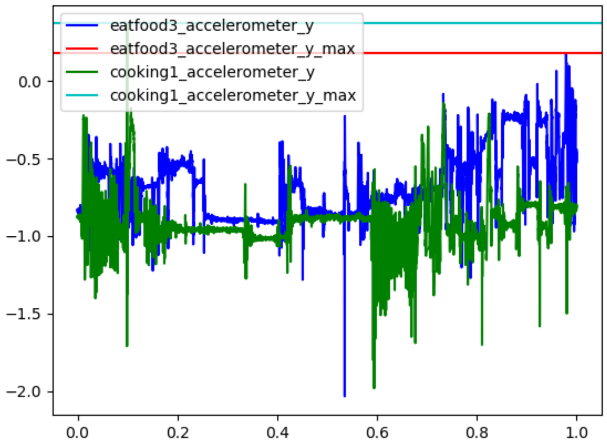
The activity of eating food would have a constant range of acceleration along the y axis while the cooking activity being more random would have a larger range of acceleration along the y axis.

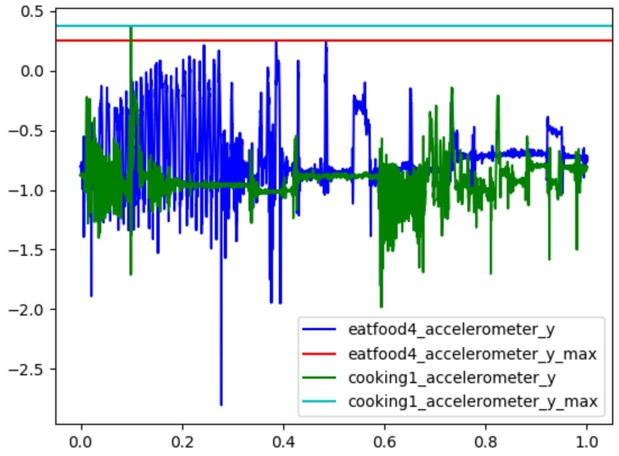
Observation:

The following plotted graphs indicate a comparison of the maximum acceleration values for each of the activities along the y axis. We can see that the maximum acceleration of cooking along the y axis is relatively higher than that of eating. **Hence, the intuition holds True.**









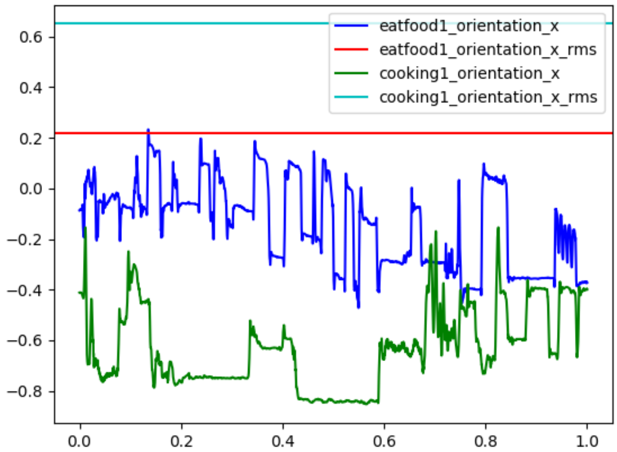
6. Root mean square of Orientation along X-axis:

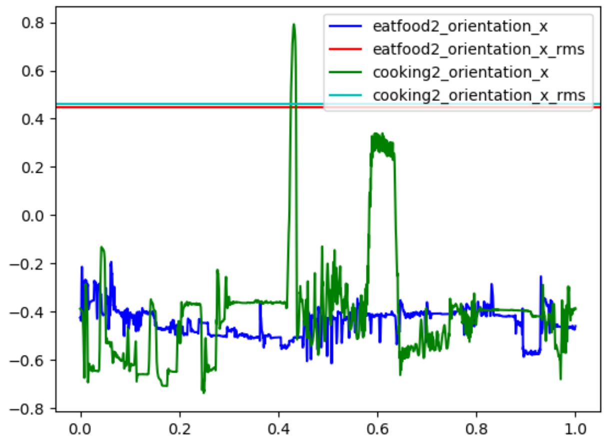
Intuition:

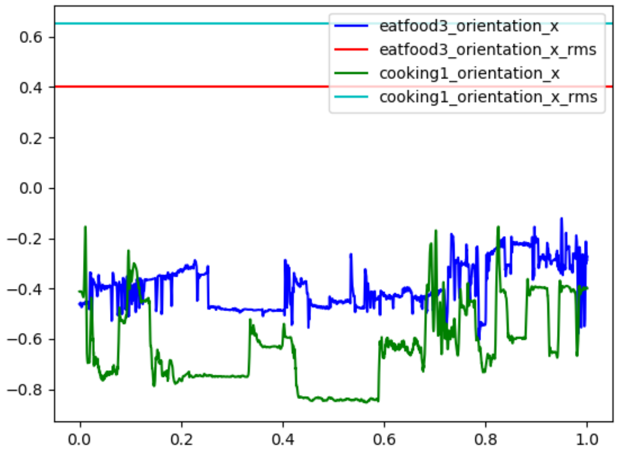
The eating activity would have a lower range of orientation changes along the x axis as compared to the cooking activity which would involve more orientation changes and would cover a larger set of values.

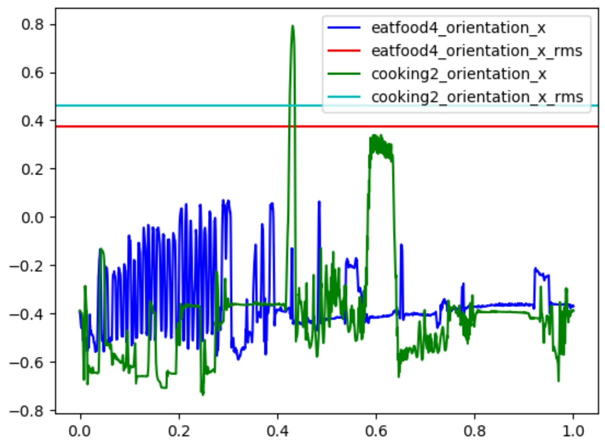
Observation:

The given plots show the rms value for the eating activity compared with the cooking activity. It can be seen that the cooking activity has a larger rms value as compared to the eating activity. **Hence, the intuition holds True.**









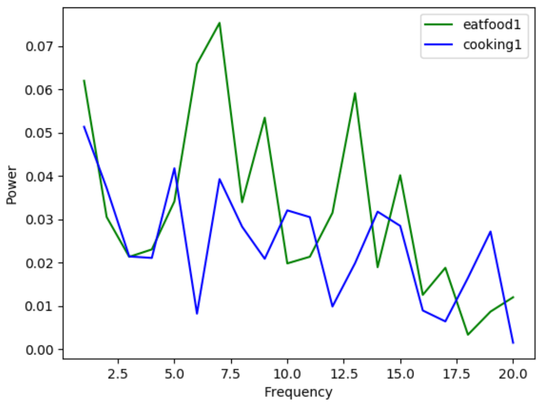
7. DFT of Acceleration along Y-axis:

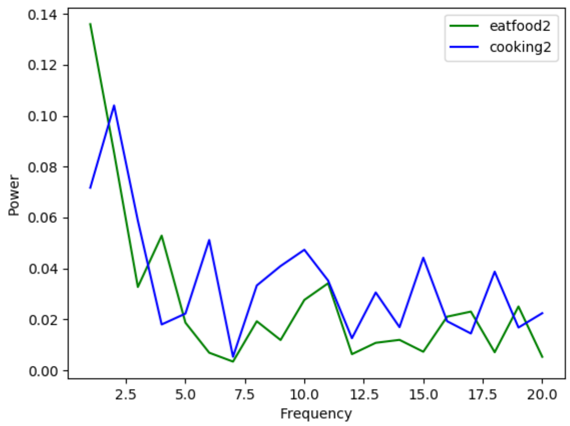
Intuition:

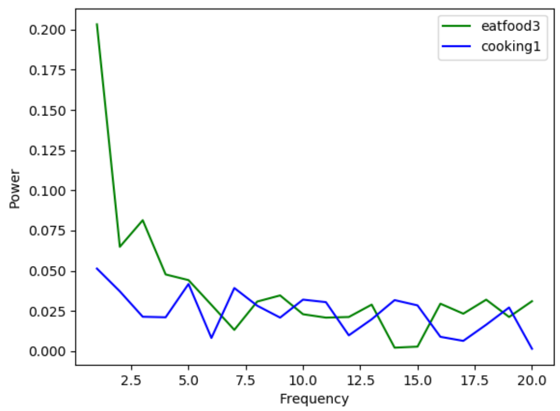
The activities of eating and cooking would be made up of multiple waves of various frequencies. These waves would be made up of different frequencies each with different energy values. We can plot the frequency vs energy graph of each activity and compare them to find uniqueness for a particular activity.

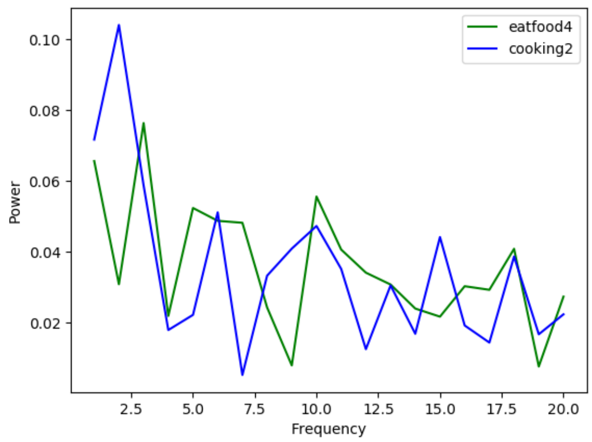
Observation:

The following plots shows the output of executing the FFT algorithm. The first 20 frequencies are plotted after excluding the 0th frequency. It is observed that the 4th and 17th frequencies have larger energy for the eating activity as compared to the cooking activity. Hence, we chose these two frequencies as **two different features** for distinguishing between eating and cooking activities.

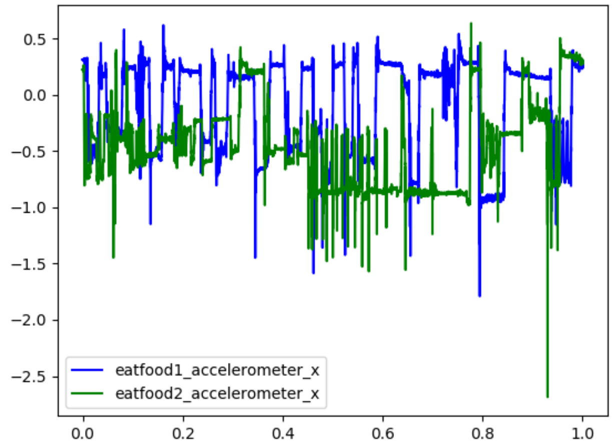


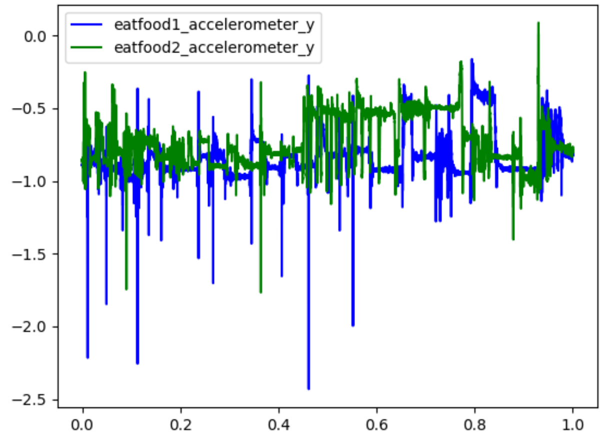


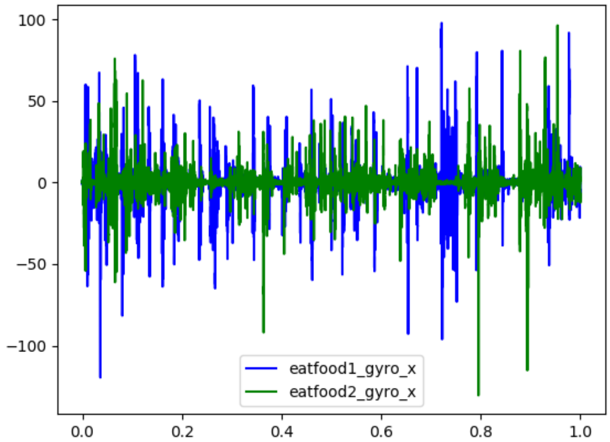


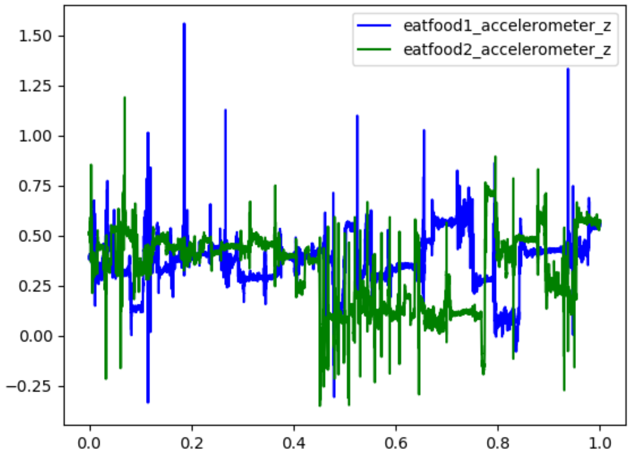


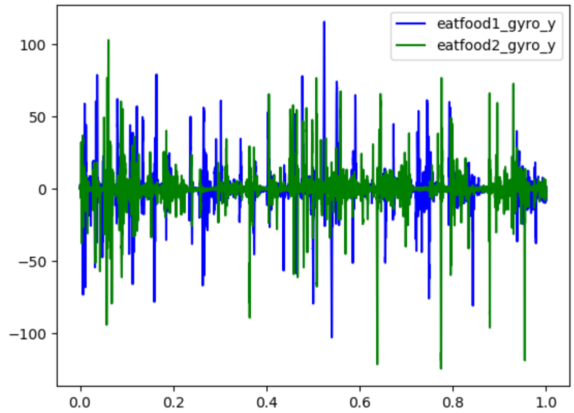
Following are the plots of each activity overlayed over another activity of the same type. Eg. Eatfood1 vs Eatfood2.
The plots were observed to find any uniqueness and patterns in the data for determining the features mentioned above.

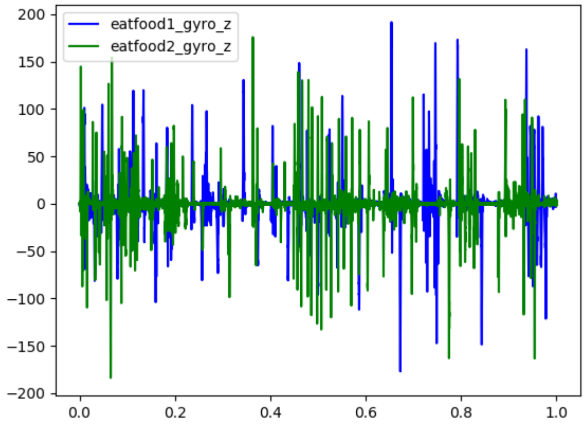


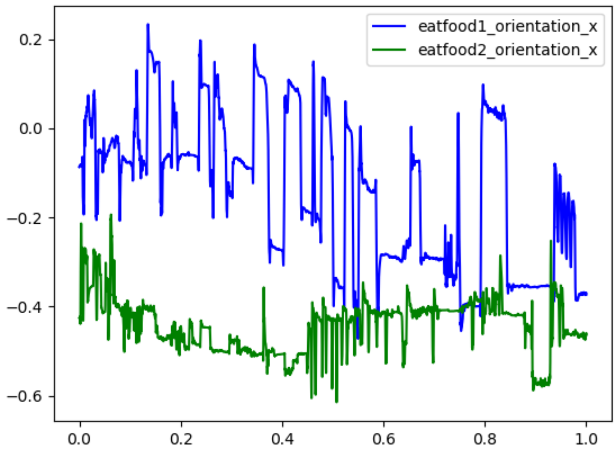


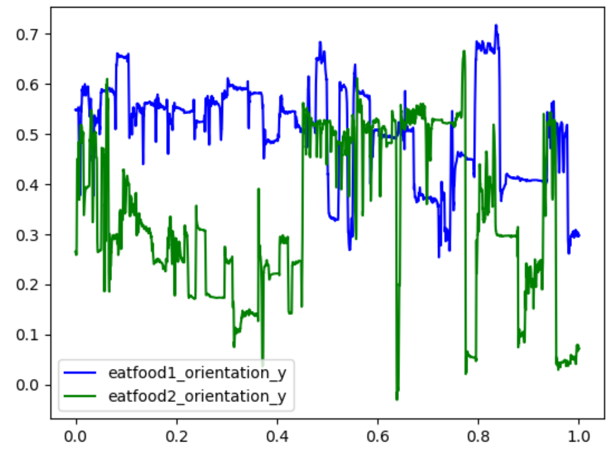


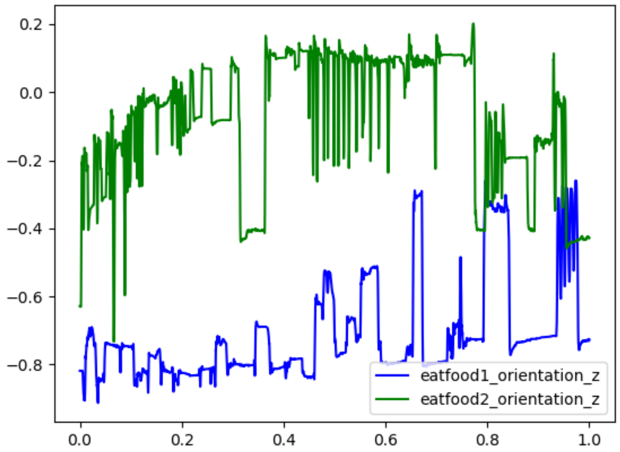


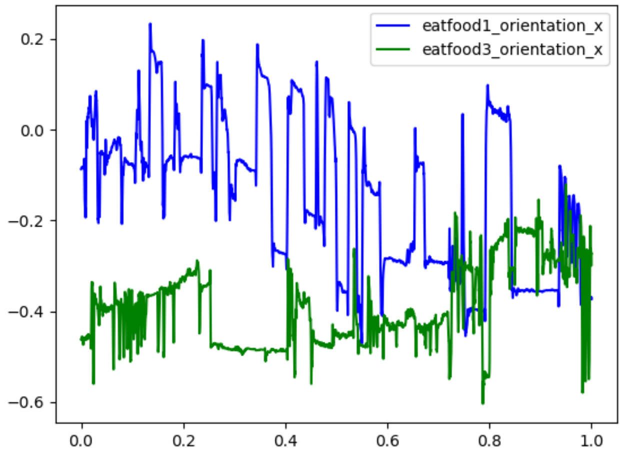


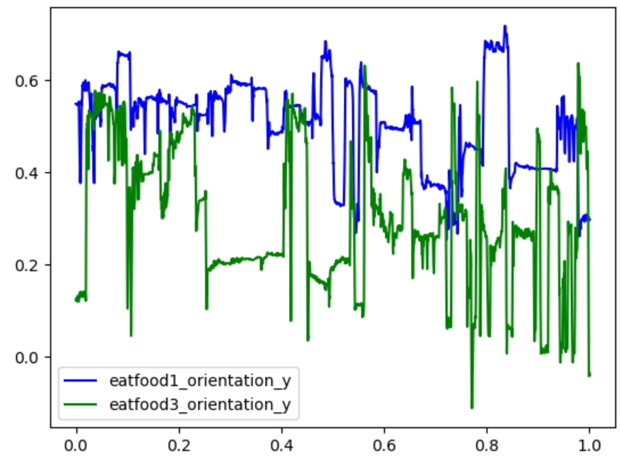


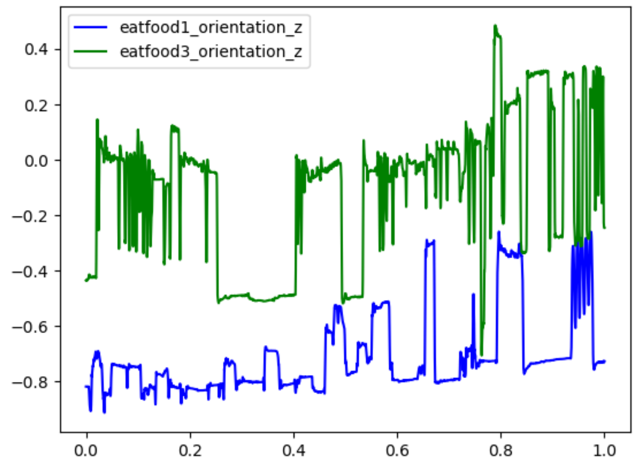

eatfood1_orientation_x
eatfood2_orientation_x
0.2
0.0
−0.2
−0.4
−0.6
0.0
0.2
0.4
0.6
0.8
1.0

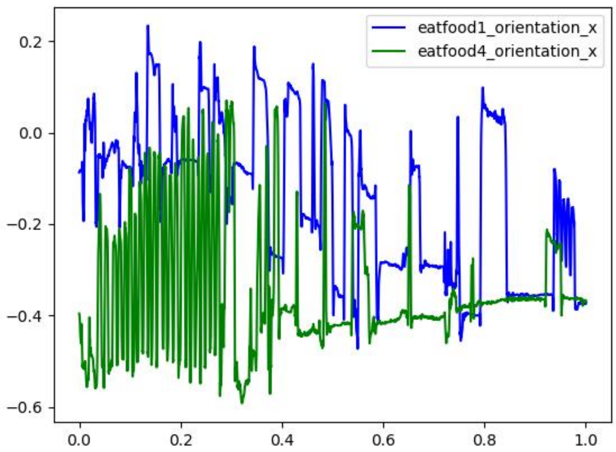

eatfood1_orientation_y
eatfood2_orientation_y
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
0.8
1.0

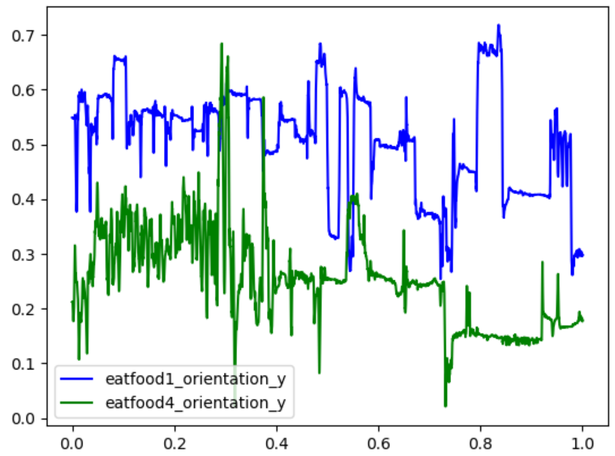

eatfood1_orientation_z
eatfood2_orientation_z
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0.0
0.2
0.4
0.6
0.8
1.0

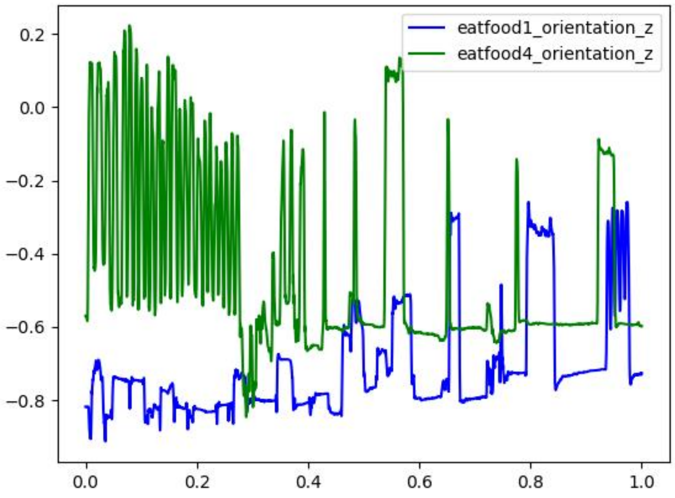

eatfood1_orientation_x
eatfood3_orientation_x
0.2
0.0
−0.2
−0.4
−0.6
0.0
0.2
0.4
0.6
0.8
1.0


eatfood1_orientation_y
eatfood3_orientation_y
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


eatfood1_orientation_z
eatfood3_orientation_z
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0.0
0.2
0.4
0.6
0.8
1.0


eatfood1_orientation_x
eatfood4_orientation_x
0.2
0.0
−0.2
−0.4
−0.6
0.0
0.2
0.4
0.6
0.8
1.0


eatfood1_orientation_y
eatfood4_orientation_y
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
0.2
0.4
0.6
0.8
1.0


eatfood1_orientation_z
eatfood4_orientation_z
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0.0
0.2
0.4
0.6
0.8
1.0

Phase 3 : Feature Selection

This phase involves reducing the dimension of the data-feature matrix and selecting top-k latent semantics which shows maximum variance among the different activities.

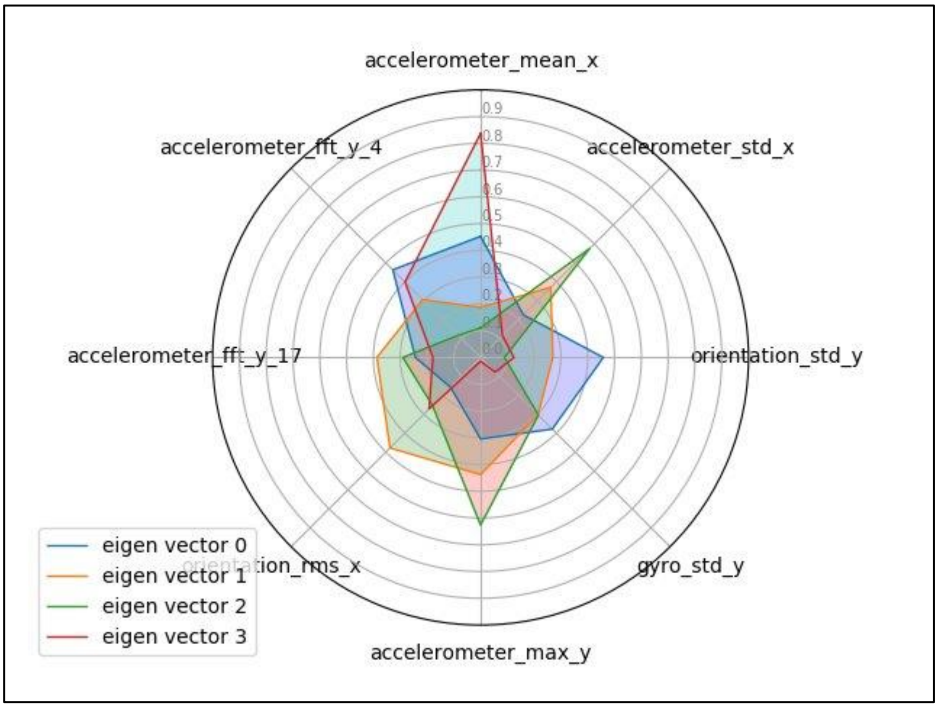
We have implemented PCA on a MxN Object-Feature Matrix where M is the number of Objects/Activities and N is the number of features extracted in Phase 2.

Subtask 1: Arranging the feature matrix

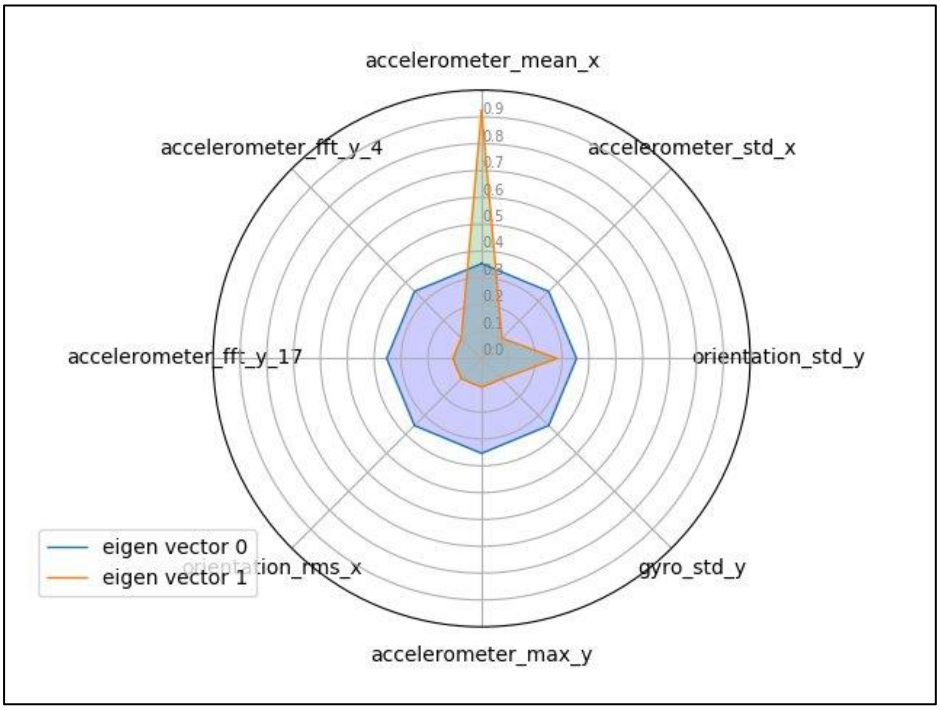
The total number of features extracted are 8. These features will become the columns of the feature matrix. Each row will correspond to each of the activities (eatfood1, eatfood2, cooking1 etc). We get 2 feature matrices, one for each type of activity (EatFood and Cooking). On applying PCA, we obtain the principal components which give us the directions along which our variance is preserved while reducing the number of dimensions.

Subtask 2: Execution of PCA

We used python's scikit-learn library to run PCA on the feature matrix generated in Phase 2. The eigen vectors generated are plotted as follows:



EatFood Activity



Cooking Activity

Subtask 3: Make sense of the PCA eigen vectors

The respective values of an eigen vector define the contribution of a feature towards the direction of the new principal component defined by the eigen vector. Since our feature matrix has negative values, our eigen vectors have negative values so we take the absolute values of the terms in our eigen vectors.

The eigen vectors obtained for the eating activity after performing PCA are:

	acc std x	orientation std y	orientation rms x	acc mean x	gyro_y_std	fft5_acc_y	fft17_acc_x	acc max y
0	0.225434	0.458041	0.157759	0.4525	0.378481	0.463287	0.243315	0.304383
1	0.370312	0.269858	0.478161	0.187291	0.302654	0.305672	0.386969	0.437665
2	0.576501	0.089356	0.253739	0.110642	0.304949	0.11757	0.290164	0.626119
3	0.633917	0.089397	0.496199	0.088965	0.080831	0.358137	0.442464	0.073886

The eigen vectors are sorted from top to bottom by their eigen values. The percentage of variance preserved by each of these are 55.207%, 39.172%, 5.621% and ~0%

We can observe that each of the features have some significant amount of contribution towards the new principal components defined by our eigen vectors. This implies that all our features are important in categorizing the activity and there is no redundancy among the features.

The eigen vectors obtained for the cooking activity after performing PCA are:

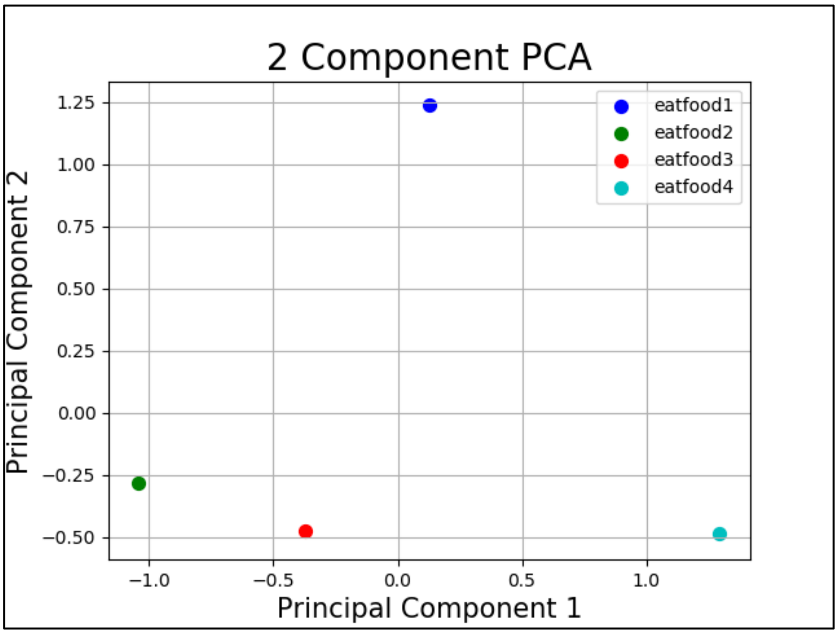
	acc std x	orientation std y	orientation rms x	acc mean x	gyro_y_std	fft5_acc_y	fft17_acc_x	acc max y
0	0.353553	0.353553	0.353553	0.353553	0.353553	0.353553	0.353553	0.353553
1	0.935414	0.133631	0.133631	0.133631	0.133631	0.133631	0.133631	0.133631

The eigen vectors are sorted from top to bottom by their eigen values. The percentage of variance preserved by each of these are ~100% and ~0%.

We can observe that each of the features have an equal amount of contribution towards our principal components along the first eigen vector which preserves almost 100% of the variance. This implies that all the selected features are important for our activity and there is no redundancy among the features.

Subtask 4: Results of PCA

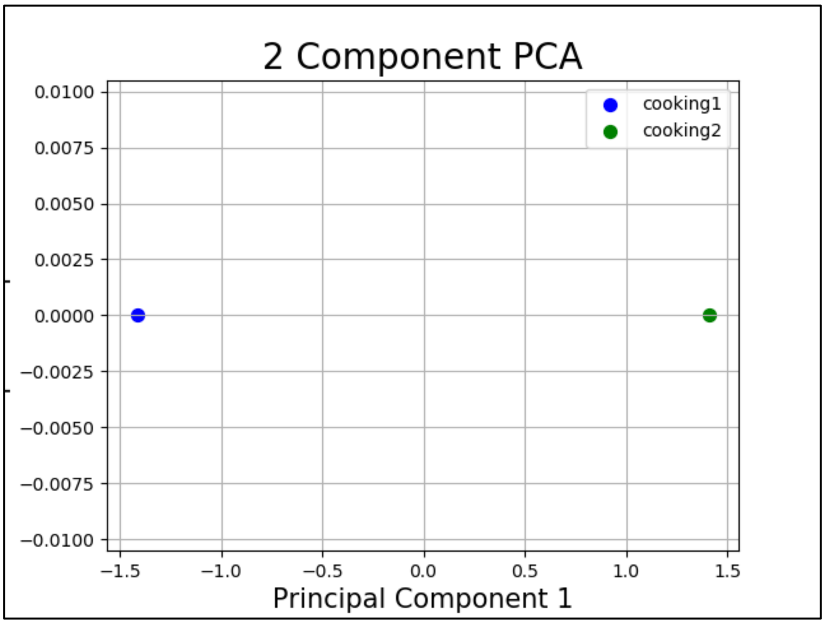
After performing PCA, our original feature matrix gets reduced to a lower dimensional matrix with the variance preserved.
For the eating activity we choose the top 2 principal components that preserve 94.379% of the variance.



Reduced Eatfood Feature matrix after applying PCA

	Principal Component 1	Principal Component 2
eatfood1	0.125545	1.237614
eatfood2	-1.04586	-0.28132
eatfood3	-0.371	-0.47241
eatfood4	1.291317	-0.48389

For the cooking activity our first principal component preserves almost 100% of the variance which can be observed in the plot below. The data can be plotted onto a single line that passes through both the points.



Reduced Cooking feature matrix after applying PCA

	Principal Component 1	Principal Component 2
cooking1	-1.41421	2.94E-16
cooking2	1.414214	2.94E-16

Subtask 5: Argue whether doing PCA was helpful or not.

In terms of selecting features, PCA did not prove to be helpful since all the selected features seem to have some contribution towards our principal components.

In terms of reducing dimensions, PCA was helpful in bringing the dimensions down from 8 to 3 in the case of the eating activity while preserving almost 100% variance. And in the case of cooking activity, it brought down the dimensions from 8 to 1 while preserving almost 100% variance.

Code in file

i. **generate_features.py**

The generate_features.py file processes the respective data files and extracts to respective features and arranges them into a matrix for each of the activities. It generates 2 csv files cooking_features.csv and eating_features.csv containing the feature matrices for each activity.

ii. **pca.py**

The pca.py file takes the generated feature matrix files and performs PCA on them. It generates the eigen vector plot as a .png file and creates 2 csv files for the eigen vectors and the reduced feature matrix.