Project Medical Wearables Report

A project on eating motion detection using two sensors
by

Priya Shukla

Under the guidance of

Prof. Dr. Walter Karlen
Tugce Canbaz Gümüssu

February 2023

Date - 20 February 2023

Table of Contents

1 Introduction

1.1 Aim of the project

The project aims to detect eating activity using machine learning with two inertial measurement units (IMU), one sensor attached to the heart level and another to the wrist.

1.2 Background

1.2.1 Quarternions

Quaternions were discovered by Sir William Rowan Hamilton (1805 - 1865), an Irish mathematician, in 1843. A quaternion is a 4-tuple, which is a more concise representation than a rotation matrix, the first non-commutative algebra to be studied.

$$i^2 = j^2 = k^2 = ijk = 1 \qquad (1)$$

To describe the rotation of the body in space we can use Quaternions and also to describe the orientation of the body in space.

1.2.2 Inertial measurement units (IMU)

IMUS Sensor provides orientation in space (Quaternions) based on IMU sensing (accelerometer, magnetometer, gyroscope). They are used in a lot of applications such as mobile phones, smart watches, virtual Reality headsets and drones. Also, they are used in the wearable medical field as they can be used in tracking activity.

1.2.3 Machine learning

According to Tom Mitchell "A a computer program is said to learn from experience E with respect to some classes of tasks T and performance measure P, if its performance at tasks in T, as measured by P improves with experience E."

Example: playing checkers. E = the experience of playing many games of checkers T = the task of playing checkers. P = the probability that the program will win the next game. Machine learning is being used almost everywhere these days. It is used in all the fields of engineering, medicine and more. Our aim is to predict the motion of eating from a trained model by machine learning.

Supervised learning

There are multiple types of learning supervised, unsupervised, reinforcement, semi-supervised etc. Supervised Learning requires labels for data as each input X requires an output Y. The model learns the pattern and predicts the output for unseen data.

Classification

We need classification algorithms to categorize multiple classes where each class has a unique feature with unique features. For example filtering a list of students on the basis of gender, weight, height or age. The number of categories in such tasks is frequently less than a few hundred, but it can be much larger and even unbounded in some difficult tasks, such as OCR, text classification, or speech recognition. Therefore, there are two kinds of classification binary and multi-class as seen in the below example fig 1.1.

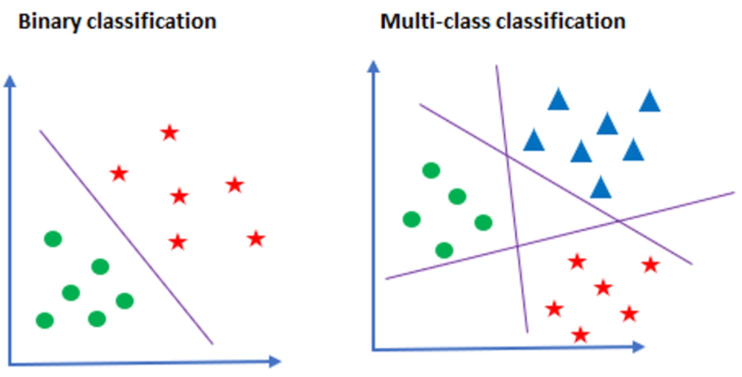


Figure 1.1: Classification

1.2.4 Classifiers

There are many types of classifiers such as logistic regression, random forest and support vector machine. Logistic regression is considered the simplest one.

Logistic regression

The central mathematical concept that underlies logistic regression is the logit—the natural logarithm of an odds ratio. The simplest example of a logit derives from a 2×2 contingency table [1]. Logistic regression is used in binary classification however linear regression is used for a continuous predication. Logistic regression is built up on linear regression but the logistic regression has sigmoid function to convert the continuous output range that came from linear regression into a binary output. Logistic regression works well when the data set is linearly separable. However if the data is not linearly separable the accuracy drops dramatically.

Support vector machine (SVM)

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection [2]. It is effective in high dimensional spaces. It is also memory efficient since it uses a subset of training points in the decision function (called support vectors). We need to kernel function like linear, radial basis function etc. SVM works well when the features can separate between the classes well.

Random Forest

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. Also, it has constraints on its depth and number of leaves also it takes its final decision depending on the majority vote.

1.2.5 Machine learning classification pipeline

Machine learning classification pipeline consists of five steps [3]. First step is collecting data from the sensor then, the data is prepossessed. There are many prepossessing ways for the data such as data cleaning, standardization and normalization. After the prepossessing step the segmentation step should be done where the segment size should be determined. Then, Features are extracted from the segments trying to distinguish between the classes. There many features can be extracted such as mean, variance and root mean square or maybe the model needs more complex feature like Spectral Entropy and Log FFT frequency bands. Finally a selected classifier is used to classify different categories.

2 Methods

2.1 Collecting Data

The data is collected using two inertial measurement unit (IMU) sensors one attached to the wrist while the other to the heart level. The IMUs are providing data with multiple columns like w,x,y,z, inclination accuracy, magnetic field distortion, vertical magnetic field, heading unreliable, timestamp, computer time etc. The data has Quaternions(w,x,y,z) from the accelerometer, magnetometer, and gyroscope from the IMUs. A Bluetooth application is installed and using Matlab we connect our IMUs to Matlab and collect the data. The raw format of received data is a text format which needs to be converted to excel later. For eating different kinds of data are collected like dinner, dinning with watching Netflix, and short eating sessions like dessert a swiss roll, yoghurt etc. For non-eating samples data from walking, working on a laptop and cleaning like wiping are collected. Also, the sample is collected at different places and in different positions. The collected data has a limitation as it lacks ground truth for verification. There are no recorded videos to label the data so we had to rely on our foreknowledge for the recorded situations if it is eating or non-eating situations. In addition, the collected data assumes that each recorded situation is either an eating or non-eating the situation, but it is unclear if the participant may have stopped eating for a short period during the recorded situation. However, all these limitations were considered while collecting the data. As the recordings for eating situations are exactly started when the participants started eating. Also, the participants are asked to try not to stop eating while the recordings are running. In addition, for the non-eating situation, the participants are asked not to eat during the recording. Finally, the collected data consists of 8 eating situations and 4 non-eating situations.

2.1.1 Preprocessing

The data contains a column as an estimation warning that tells us whether the accuracy of the estimations is reliable or not. Hence we check the column for the warning and drop the data when it is not reliable. Since we use two sensors we also need to drop data for the other sensor when one of the sensor's data is not reliable. Also, we use the tics from the timestamp which is the time from the sensor. Therefore, we make a linear loop of time from the circular loop.

We take one of the sensor's timestamps and do the relative counter starting from zero and keep adding the tics from the different array of two consecutive rows of timestamps and make sure the counter keeps going even after 256. We also add a difference value of 256 and the last value before the counter sets to zero to maintain the relative tics accurate. Finally, we check the size of the datasets from two sensors and trim them according to the length of the shortest sample. An average movement filter is also used to filter the noise and make the movement more smooth over time.

2.1.2 Data cleaning

The data cleaning part is performed after the feature extraction step. It gave good results. As the sensor gave us the orientation in space and the relative position is extracted from it. Hence, the data cleaning the step is done after the feature extraction step, as this would make the model more robust to non-intentional movements occurred during the recordings. Since there is no ground truth, the entire recording was labelled as either eating or non-eating based on prior knowledge. Only the training set is cleaned, while the test set was left unchanged to ensure that the model is tested correctly. The outliers were spotted and their values were replaced by the median as the median is more robust against the extreme values. The Interquartile range method was used to spot the outliers. Also, the percentage of the outliers was checked as it has to be around five per cent. Finally, a box plot was plotted for each feature.

2.1.3 Relative position Extraction

We have two sensors one for the reference point attached to the chest and another on the writ to detect the motion of eating. We use the relative position of the wrist. That is we always find the difference in quaternion values between both the sensors. This way the minimal could be found easily as the chest is near the mouth. Though using two sensors is more work than one sensor we try to study the trained model generated by machine learning for this situation. We calculate the relative position for all four variables w, x, y, and z by subtracting both the values from both the sensors.

$$W_{relative} = W_{chest} - W_{wrist} \tag{2}$$

$$X_{relative} = X_{chest} - X_{wrist} \tag{3}$$

$$Y_{relative} = Y_{chest} - Y_{wrist} \tag{4}$$

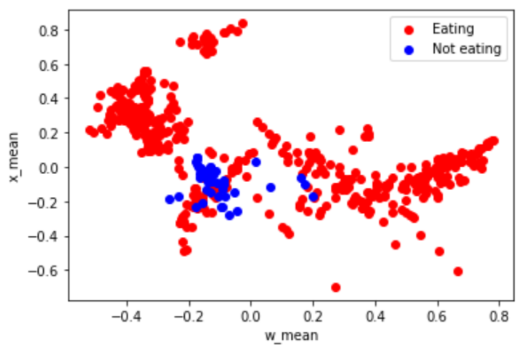
$$Z_{relative} = Z_{chest} - Z_{wrist} \tag{5}$$

2.1.4 Frame and Feature extraction

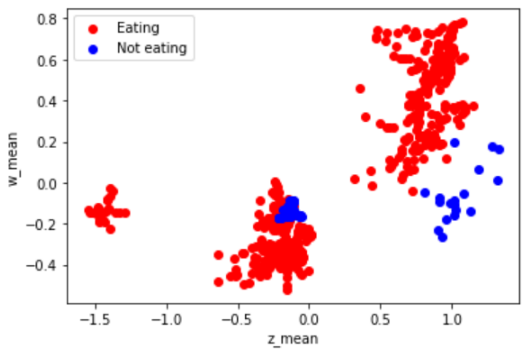
We have 14.9831 ticks of the sensors is equivalent to 1s from data calculation. We need to make frames with 7s i.e., 104.8817 ticks. We chose the frame size wisely and after reading some literature 7s was chosen to be the best frame size. However, we deal with a timestamp from sensors so we need some calculation and we found that we have 14.9831 ticks of the sensors is equivalent to 1s. We need to make frames with 7s i.e., 104.8817 ticks. However, it s not accurate because we couldn't find this data from the datasheet of the sensor and we can also play around a bit with changing the values of the tics and find the best possible frame. The size of the window depends on the duration of the event wanted to be detected. For the targeted case the frame size should take the entire food intake gesture. The frames were extracted using a traditional sliding window approach with a 50 per cent overlap. Intake gesture duration is affected by many factors such as the type of food, whether the individuals are hungry or not, the eating style itself and whether they are multi-tasking or not. Based on the observed data in work [4], the intake gesture period ranges from 2 second to 10 seconds. Also, it is said that the best size of the window is around the midpoint of this range which is 6 seconds at this point they achieved the best classification results. Five statistical functions for each frame were computed which are the mean, variance, skewness, kurtosis, and root mean square as in work [4]. As each frame has four axes and these functions were applied for each axis in each frame so the feature extraction output was 20 features.

2.1.5 Features investigations

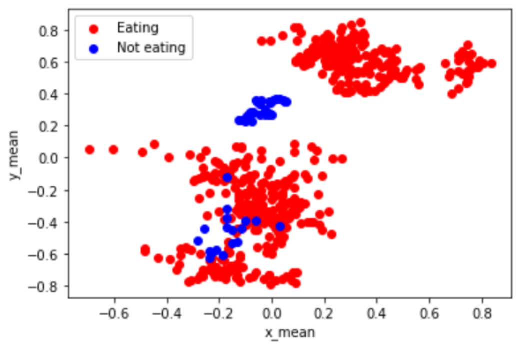
We have rms, mean, variance., skewness and kurtosis features for each axis. Therefore, we have 20 features. Since kurtosis is similar to skewness we drop these features and only proceed with 16 features. Furthermore, we try to find the combinations of features that are capable of differentiating between the two classes which are eating and non-eating. from the below figures 2.1, it is better to use the mean feature over others as shown in figures 2.2, 2.3, 2.4, and2.5. However, the mean and skewness plot and variance feature graph shows it as



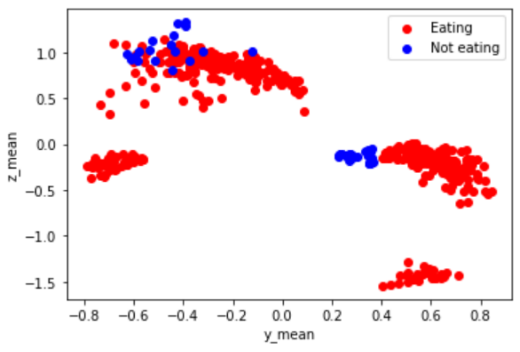
(a) Combination between X and W axes



(b) Combination between W and Z axes



(c) Combination between Y and X axes



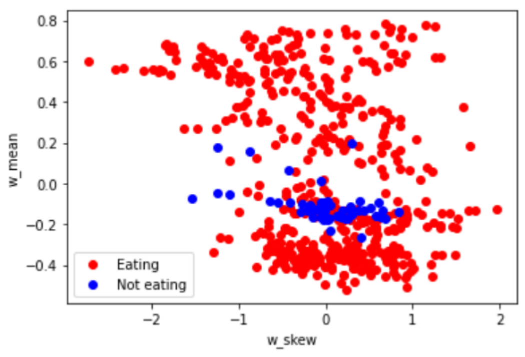
(d) Combination between Z and Y axes

Figure 2.1: combination between different axes for mean feature

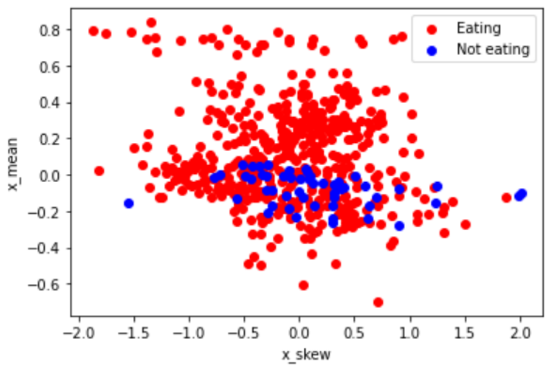
a bad feature. in Figures 2.2 and 2.4.

2.1.6 Classification

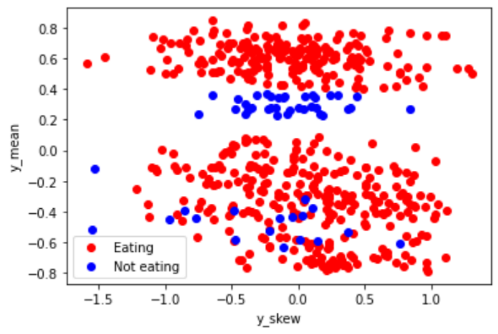
We perform binary classification to differentiate our two events: Eating Gesture Eating Moment The classification is done on the basis of the features mean, variance, rms, and skew for each frame. In our work, a classifier was used, which is based on the chosen features which are the mean, variance, rms, and skew to determine for each frame whether it is an eating gesture or not. The classifier classifies each n-tics(frame size) as either an eating or non-eating gesture, and then the density of the eating gestures over a certain period of time was calculated. Ideally with our tics calculation 14.9831 ticks of the sensors are equivalent to 1s. We need to make frames with 7s i.e., 104.8817 ticks. But we



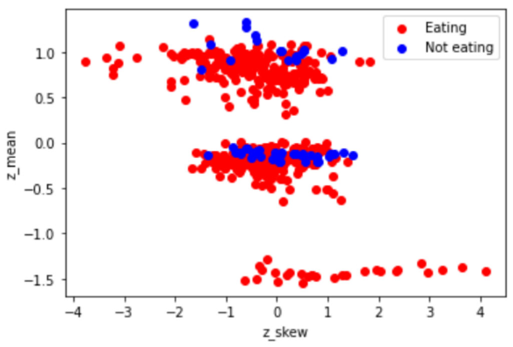
(a) W axis mean and skew



(b) X axis mean and skew

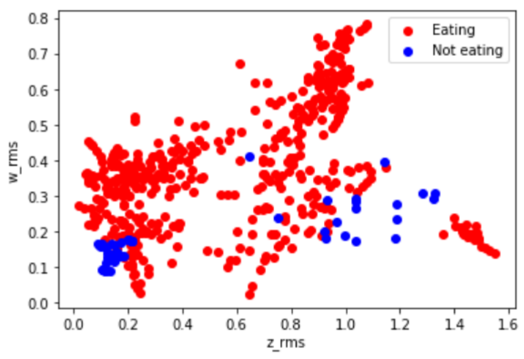


(c) Y axis mean and skew

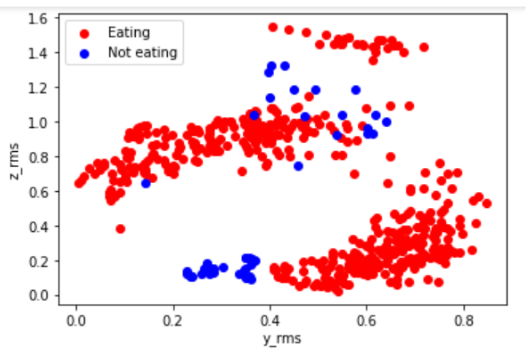


(d) Z axis mean and skew

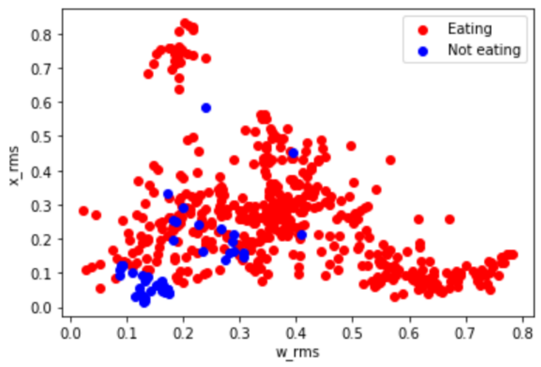
Figure 2.2: combination between different axes for mean and skewness



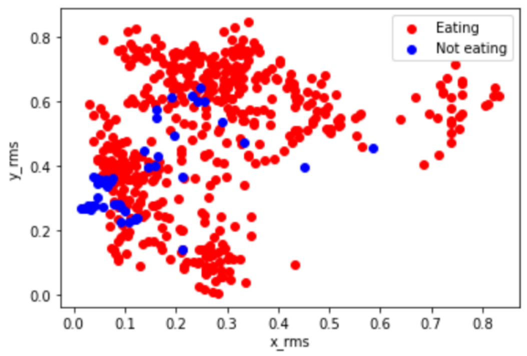
(a) Combination between W and Z axes



(b) Combination between Z and Y axes

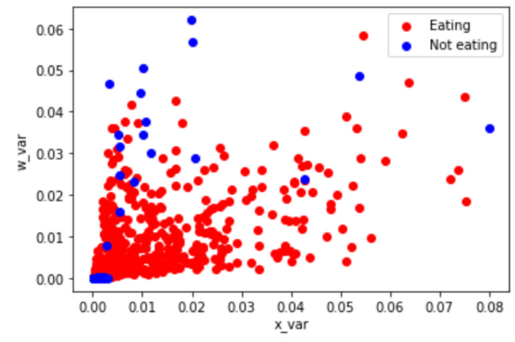


(c) Combination between X and W axes

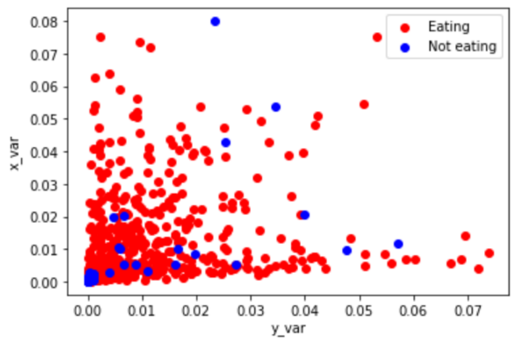


(d) Combination between Y and X axes

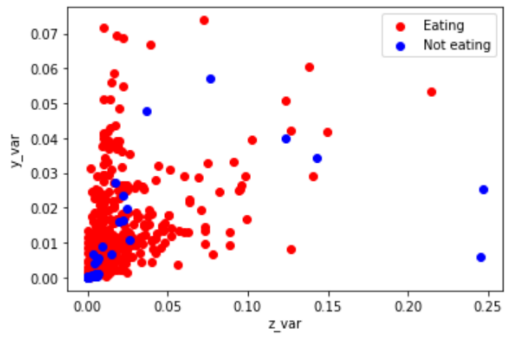
Figure 2.3: combination between different axes for rms feature



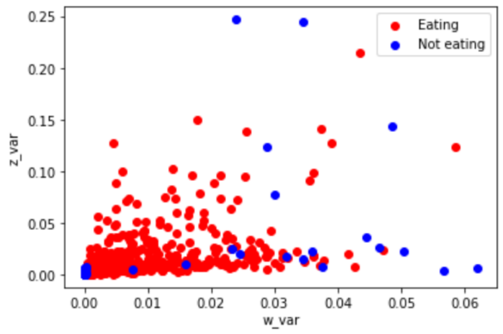
(a) Combination between W and X axes



(b) Combination between X and Y axes

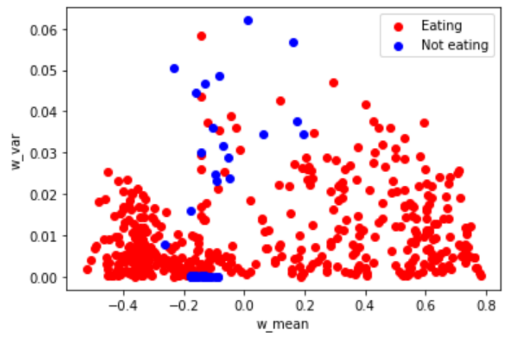


(c) Combination between Y and Z axes

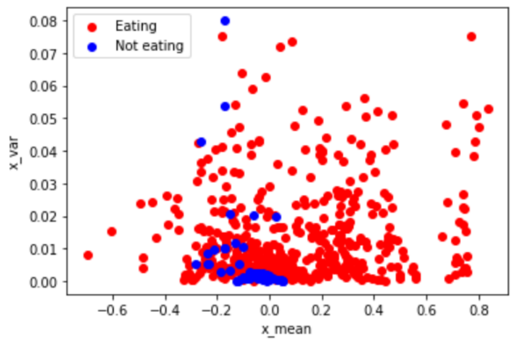


(d) Combination between Z and W axes

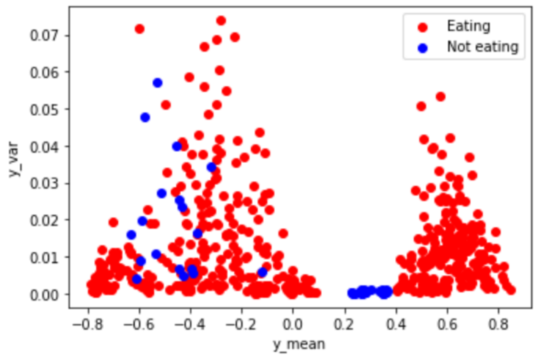
Figure 2.4: combination between different axes for var feature



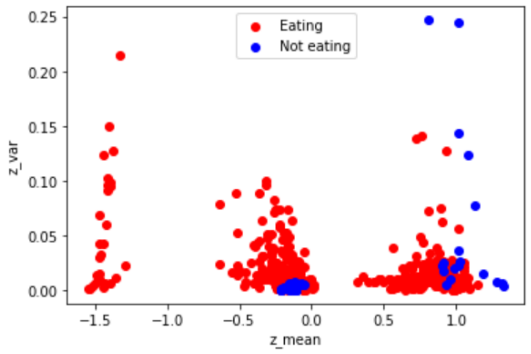
(a) W axis mean and var



(b) X axis mean and var



(c) Y axis mean and var



(d) Z axis mean and var

Figure 2.5: combination between different axes for mean and var feature

can also change these tics and see the result. Based on this density, the period is classified as an eating a moment or not. Three Classifiers for eating gestures were used and tested. The first one was the Support vector machine (SVM) with radial basis function (RBF) kernel and also with linear kernel. Also, Random forest was used, in addition to logistic regression. A function was implemented to detect the eating moment. This function uses a fixed time window of 120 s. The function calculates the density of eating gestures within 17 frames as each frame is 7 seconds and determines that a period is considered an eating moment if the density of eating gestures is greater than 70%.

3

Results

The data collection is done on ”The Hocoma Valedo Home”[reference website]. We have tried three different models. We have used different files for testing and training to test the models as an unknown scenario. Out of 12 scenarios, we use 4 for the testing part. Each recording consists of a sequence of frames that are repeated throughout the recording, making it important to test the models on new data to obtain a more accurate evaluation of their performance. For the testing, we have two non-eating scenarios and 2 eating scenarios hence making up to 50%. Also, we have tried to train our model with the mean feature and also without the mean feature. We have got different performances in both situations.

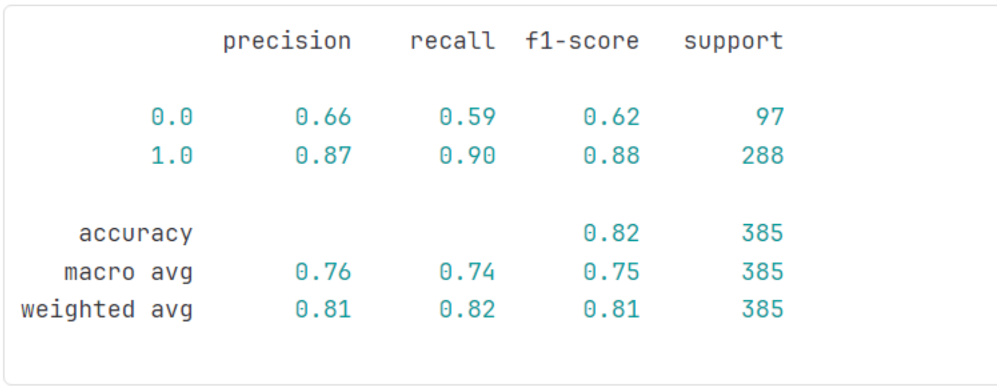
3.0.1 Support vector machine (SVM)

The accuracy of the Support Vector Machines (SVM) for classifying eating gestures was around 81% and for detecting eating moments was 96%, when we trained the model without mean feature and the best kernel is rbf with Penalty parameter of the error term of 100. For a better understanding of the performance of our model, the confusion matrix is shown in figure 3.1 and Classification report is shown in figure 3.2 .

However, when we train the model with mean feature the accuracy for classifying eating gestures was around 79.8% and detecting eating moments was 95.6%. In this situation, the best-performing kernel is linear with a Penalty parameter of the error term of 100. For a better understanding of the performance of our model, the confusion matrix is shown in figure 3.4 and Classification report is shown in figure 3.3. However, the $y_prediction$ sample class looks far better than the model trained without mean.

3.0.2 Random forest

The accuracy of the Random forest for the trained model without mean features for classifying eating gestures was around 76.6% and for detecting eating moments was around 92.3%. the confusion matrix of the model is shown in figure 3.6 and the Classification report is shown in figure 3.5. However, when we trained the model with mean features it performed really badly. Accuracy for classifying eating gestures was around 61.5% and for detecting eating moments was around 65.2%. the confusion matrix of the model is shown in figure


```
              precision    recall  f1-score   support

         0.0       0.66      0.59      0.62        97
         1.0       0.87      0.90      0.88       288

    accuracy                           0.82       385
   macro avg       0.76      0.74      0.75       385
weighted avg       0.81      0.82      0.81       385
```

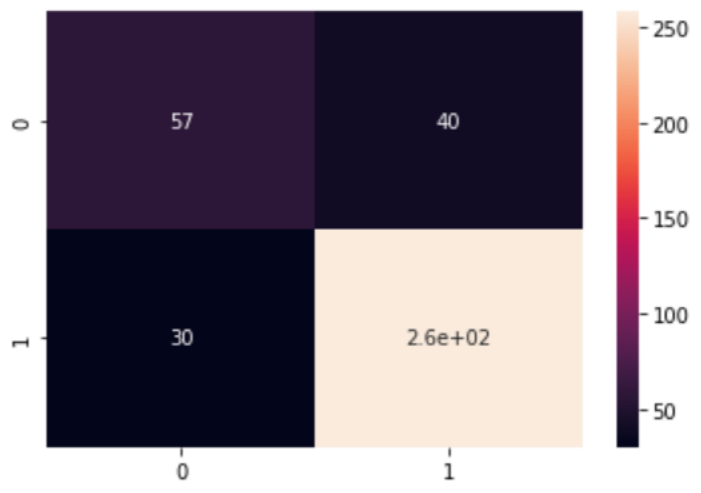


Figure 3.1: Confusion Matrix without mean feature for SVM, kernel=rbf, C=100

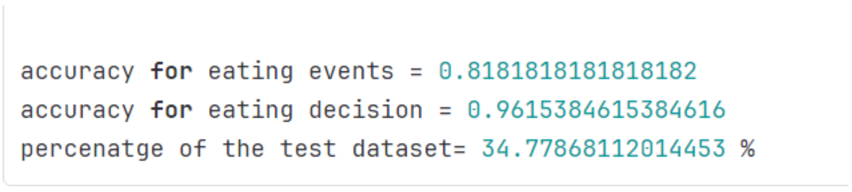

```
accuracy for eating events = 0.8181818181818182
accuracy for eating decision = 0.9615384615384616
percenatge of the test dataset= 34.77868112014453 %
```

Figure 3.2: Classification report without mean feature for SVM, kernel=rbf, C=100

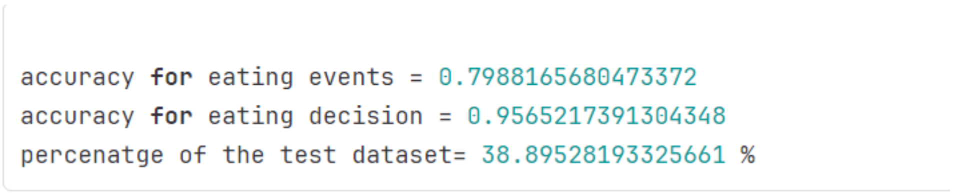
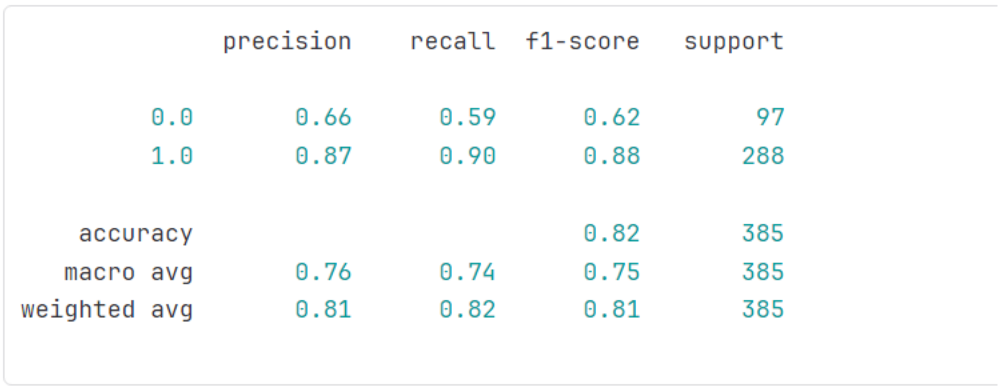

```
accuracy for eating events = 0.7988165680473372
accuracy for eating decision = 0.9565217391304348
percenatge of the test dataset= 38.89528193325661 %
```

Figure 3.3: Classification report with mean feature for SVM, kernel=linear, C=100



```
              precision    recall  f1-score   support

         0.0       0.66      0.59      0.62        97
         1.0       0.87      0.90      0.88       288

    accuracy                           0.82       385
   macro avg       0.76      0.74      0.75       385
weighted avg       0.81      0.82      0.81       385
```

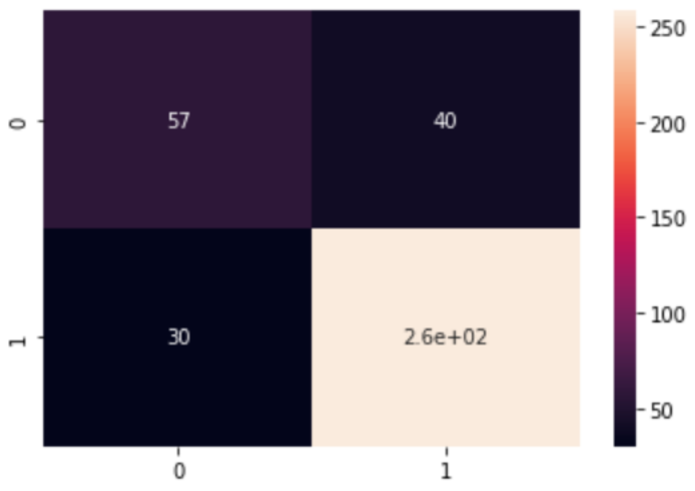


Figure 3.4: Confusion Matrix with mean feature for SVM, kernel=linear, C=100

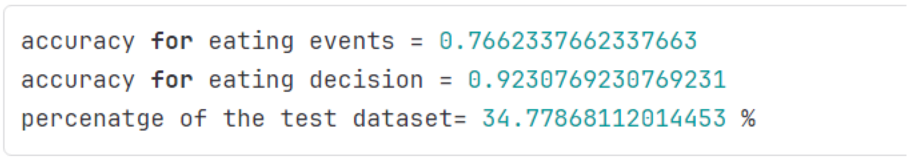
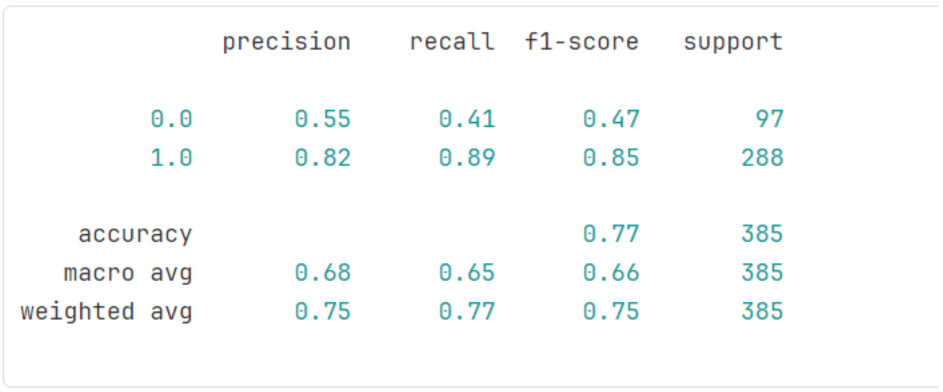

```
accuracy for eating events = 0.7662337662337663
accuracy for eating decision = 0.9230769230769231
percenatge of the test dataset= 34.77868112014453 %
```

Figure 3.5: Classification report without mean feature for Random Forest

3.8 and the Classification report is shown in figure 3.7.

3.0.3 Logistic regression

The accuracy of the Logistic Regression for the trained model without mean features for classifying eating gestures was around 75.5% and for detecting eating moments was around 76.9%.in figure 3.10 the confusion matrix and 3.9 the Classification report of the model were printed respectively. However, for the model trained with mean features the accuracy for classifying eating gestures was around 65.6% and for detecting eating moments was around 69.5% in figure 3.11 the confusion matrix and the Classification report 3.12 of the model were printed respectively.



```
              precision    recall  f1-score   support

         0.0       0.55      0.41      0.47        97
         1.0       0.82      0.89      0.85       288

    accuracy                           0.77       385
   macro avg       0.68      0.65      0.66       385
weighted avg       0.75      0.77      0.75       385
```

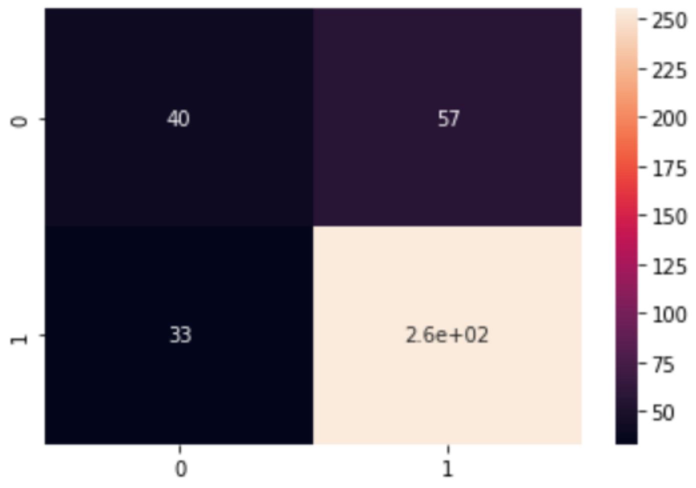
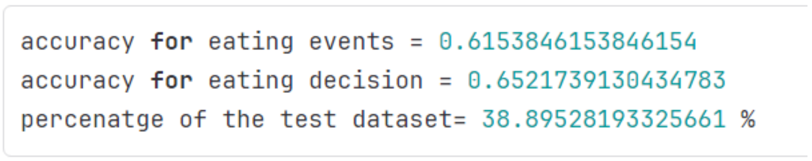
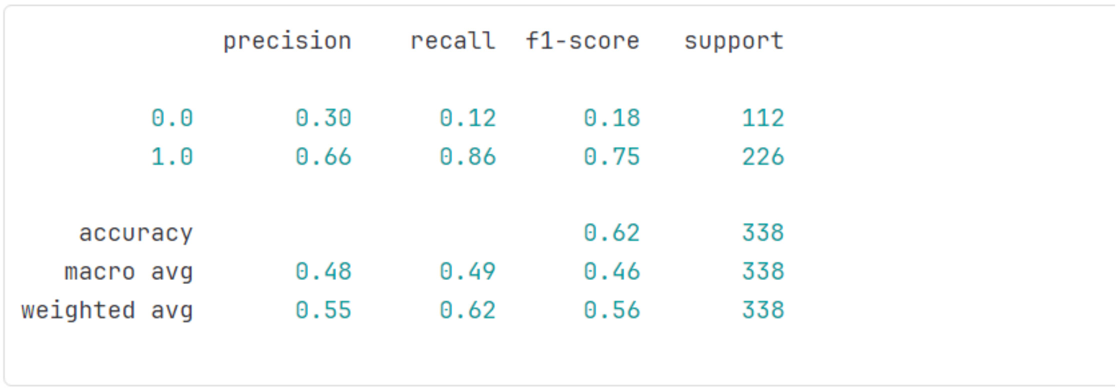


Figure 3.6: Confusion Matrix without mean feature for Random Forest



```
accuracy for eating events = 0.6153846153846154
accuracy for eating decision = 0.6521739130434783
percenatge of the test dataset= 38.89528193325661 %
```

Figure 3.7: Classification report with mean feature for Random Forest



```
              precision    recall  f1-score   support

         0.0       0.30      0.12      0.18       112
         1.0       0.66      0.86      0.75       226

    accuracy                           0.62       338
   macro avg       0.48      0.49      0.46       338
weighted avg       0.55      0.62      0.56       338
```

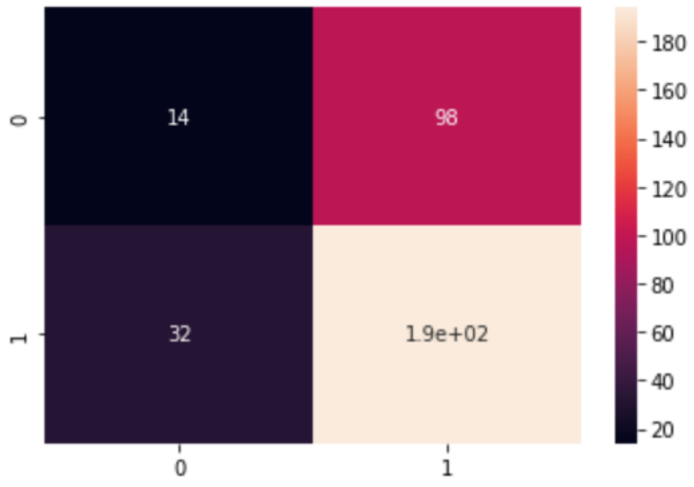
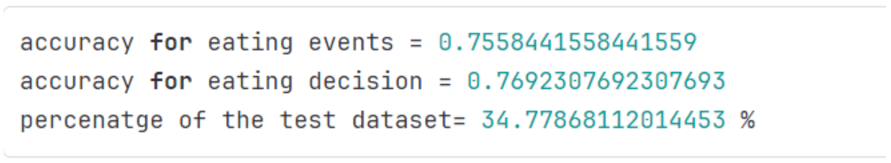
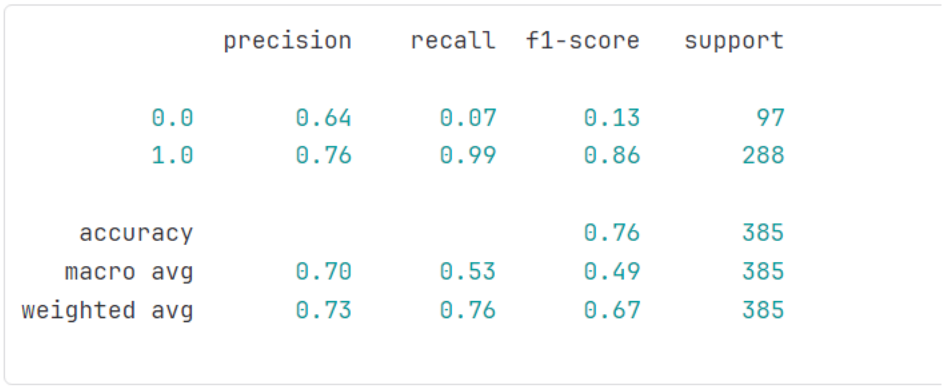


Figure 3.8: Confusion Matrix with mean feature for Random Forest



```
accuracy for eating events = 0.7558441558441559
accuracy for eating decision = 0.7692307692307693
percenatge of the test dataset= 34.77868112014453 %
```

Figure 3.9: Classification report without mean feature for logistic regression



```
              precision    recall  f1-score   support

         0.0       0.64      0.07      0.13        97
         1.0       0.76      0.99      0.86       288

    accuracy                           0.76       385
   macro avg       0.70      0.53      0.49       385
weighted avg       0.73      0.76      0.67       385
```

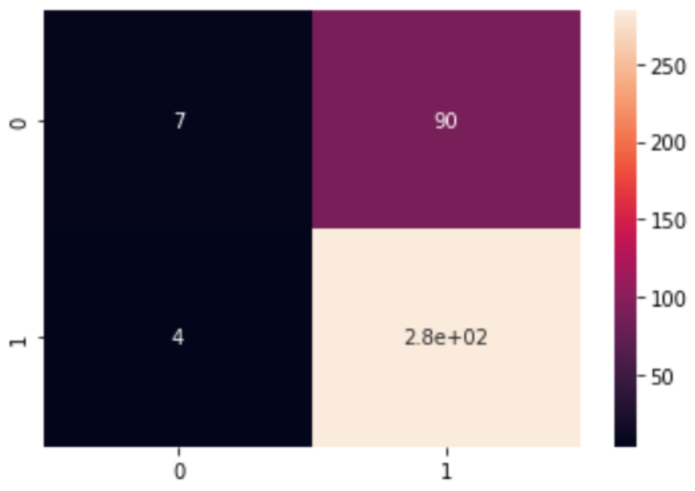
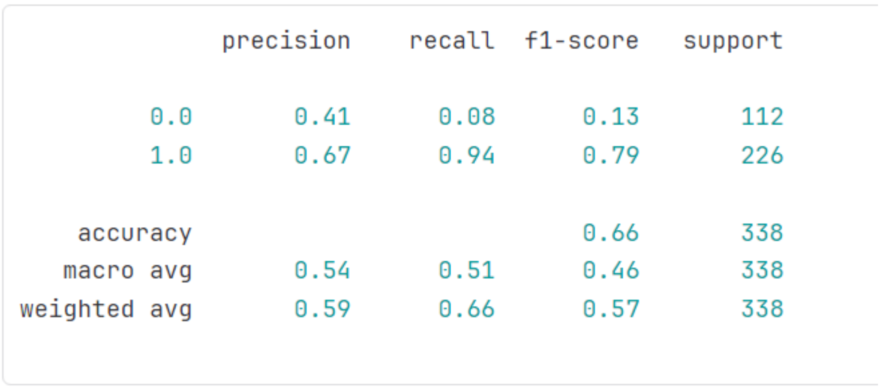


Figure 3.10: Confusion Matrix without mean feature for Logistic Regression



```
              precision    recall  f1-score   support

         0.0       0.41      0.08      0.13       112
         1.0       0.67      0.94      0.79       226

    accuracy                           0.66       338
   macro avg       0.54      0.51      0.46       338
weighted avg       0.59      0.66      0.57       338
```

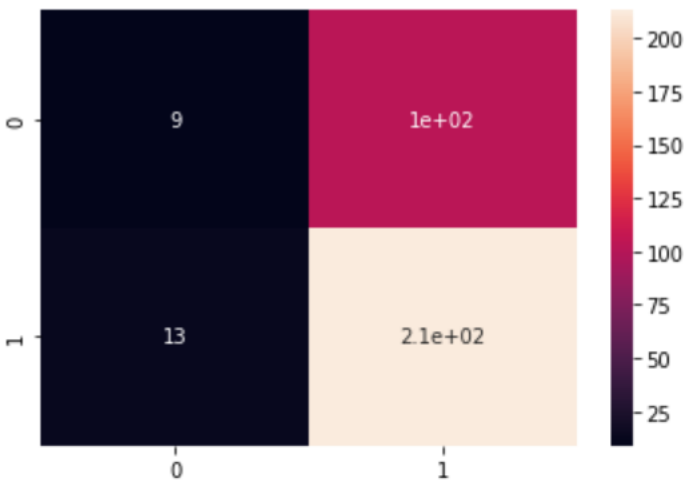
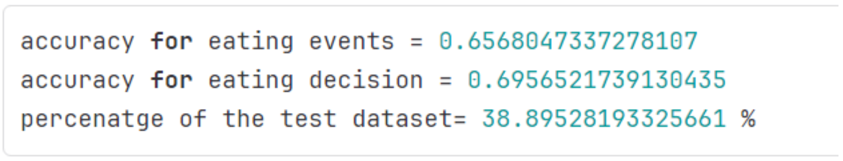


Figure 3.11: Confusion Matrix with mean feature for Logistic Regression



```
accuracy for eating events = 0.6568047337278107
accuracy for eating decision = 0.6956521739130435
percenatge of the test dataset= 38.89528193325661 %
```

Figure 3.12: Classification report with mean feature for Logistic Regression

4 Discussion

We have two trained models one with mean features and the other without mean features. In both, the models SVM has performed better than other algorithms like Random Forest and Logistic Regression. However, when the model was trained without mean features Random Forest performed average but when trained with mean features its accuracy dropped. However, for Support Vector Machine we have seen that we need Radial Basis Function(RBF) kernel for the model trained without mean features and for a model with mean features linear kernel has performed the best. Additionally, with the glance on y prediction, the model trained with mean features tends to produce better results than without mean features. Logistic regression has not produced significant accuracy for both models. The model worked differently when datasets were swapped from the training and testing set. Also changing the size of the window has affected the accuracy differently for the two models. Increasing the window frequency has dropped accuracy in the model without features but has improved in features with mean for example it performed well with 200 tics from 105. The major challenge faced was with data collection as the sensor or Matlab app or combination of both didn't respond well if recorded for over 10 mins. Overall with SVM, it gives the best results and for two sensors it may not be the best solution because the sensors need to be attached to the skin which could be a bit uncomfortable.

5 Resources

- Link to the parent folder
- Link to the dataset
- Link to code without mean features.
- Link to the code with mean features

References

1. Peng, J., Lee, K. & Ingersoll, G. An Introduction to Logistic Regression Analysis and Reporting. *Journal of Educational Research - J EDUC RES* **96,** 3–14. doi:10.1080/00220670209598786 (Sept. 2002).
2. `https://scikit-learn.org/stable/modules/svm.html`.
3. Bulling, A., Blanke, U. & Schiele, B. A Tutorial on Human Activity Recognition Using Body-Worn Inertial Sensors. *ACM Comput. Surv.* **46.** ISSN: 0360-0300. doi:10.1145/2499621. https://doi.org/10.1145/2499621 (Jan. 2014).
4. Abowd, G. D. *What next, Ubicomp? Celebrating an Intellectual Disappearing Act* in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (Association for Computing Machinery, Pittsburgh, Pennsylvania, 2012), 31–40. ISBN: 9781450312240. doi:10.1145/2370216.2370222. https://doi.org/10.1145/2370216.2370222.
5. AM. Mohri A. Rostamizadeh, A. T. *Foundations of Machine Learning* (MIT Press, 2018).