Human Activity Recognition with Smartphones

University of Malta

Faculty of Information & Communication Technology

Bachelor of Science in Information Technology (Honours) (Artificial Intelligence) 2nd Year

ICS2000 – Group Assigned Practical Task

Mr Kristian Guillaumier

|  |  |  |
| --- | --- | --- |
| Sean Farrugia | Owen Agius | Francesca Maria Mizzi |
| 258801(L) | 496701(L) | 118201(L) |

# Table of Contents

[Table of Contents 1](#_Toc72725779)

[1 - Introduction 2](#_Toc72725780)

[Distribution of Work: 3](#_Toc72725781)

[2 - Research and Literature Reviews 4](#_Toc72725782)

[2.1 - A Public Domain Dataset for Human Activity Recognition Using Smartphones 4](#_Toc72725783)

[2.2 - Training Computationally Efficient Smartphone–Based Human Activity Recognition Models 5](#_Toc72725784)

[2.3 - Energy Efficient Smartphone-Based Activity Recognition using Fixed-Point Arithmetic 6](#_Toc72725785)

[Pros and Cons 7](#_Toc72725786)

[3 - Implementation and Testing 8](#_Toc72725787)

[3.1 - Support Vector Machines 8](#_Toc72725788)

[3.1.1 - Radial Kernel 8](#_Toc72725789)

[3.1.2 - Polynomial Kernel 9](#_Toc72725790)

[3.2 – K-Nearest Neighbours 9](#_Toc72725791)

[3.2.1 - Distance Weighted k-NN 10](#_Toc72725792)

[3.3 - Decision Trees 10](#_Toc72725793)

[3.4 - Logistic Regression 11](#_Toc72725794)

[3.5 - Testing 12](#_Toc72725795)

[4 - Evaluation and Critical Analysis 14](#_Toc72725796)

[4.1 - Dataset 2: Six New Activities 15](#_Toc72725797)

[Chapter 5: Conclusions 17](#_Toc72725798)

[References 18](#_Toc72725799)

# 1 - Introduction

The main scope of this Assigned Practical Task (APT) is to implement a Human Activity Recognition System (HAR System). The purpose of this system is to identify the action which the user would be doing, solely based off of the changes in motion of the user’s body during the performance of specific actions.

Most HAR systems implemented would use specialised motion sensors that would be secured to the users’ body, including, but not limited to, the waist, chest, arms and legs. However, the main problem with this type of system is the complex setup the user would be required to wear during the activity, in addition to the added expenses when purchasing these sensors. Considering the simplicity of the application, many users are more likely to get discouraged in using such a complex, albeit excessive, set up. As a result of the rapid advancements in the technological field as well as the efforts of many researchers, this setup has been reduced to needing only a smartphone. This initial set-up made use of a bulkier mounting system through the use of a belt, an aspect of the set-up which can be improved upon, with the users’ comfort being the main priority.

Therefore, for our APT, we were aiming to develop a simple Human Activity Recognition prototype which only uses the built-in sensors found in an average smartphone and eliminating the use of a belt mount, allowing the user to carry their phone in their pockets. While this may result in less accurate predictions, it allows users to retain their usual habits; keeping their phone in their pockets.

Moreover, two separate datasets were gathered by the three members working on this APT. The first dataset was done to mimic that made in the paper [1] with six total actions: Walking, Walking Downstairs, Walking Upstairs, Sitting, Standing, and Laying. This dataset was created in order to compare the difference in results gathered and processed by Anguita et al. and ourselves. However, we also collected a second dataset in which we chose physical activities which were not included in the existing data. These also required body movement from the user and were recorded through the accelerometer and gyroscope sensors found in the smartphone. The final new activities are Cycling, Football, Swimming, Tennis, Jump Rope and Push-ups. In summary, the main aim of this project was not only to interpret the original six activities that most Human Activity Recognition papers tend to focus on, but also to recognise another six unique physical activities.

The process of classifying the data with a high accuracy can be divided into two steps: data collection and modelling. In order to collect a sufficient amount of data, the free app ‘Androsensor’ was used. Using this app allowed for the collection of data using the four main inertia sensors: gyroscope, gravity, accelerometer and linear acceleration. The data collected consists of roughly one hour worth of data for each of the 12 activities mentioned above, allowing for the model to be developed with an even distribution of data across all the categories.

After the data is collected, it is pre-processed. The pre-processing entails the removal of “NaN” and duplicated values while also generating statistical readings from the ‘csv’ file produced by ‘Androsensor’. After being processed, the data is analysed via a t-SNE algorithm which aids the visualisation of data clusters. Finally, the data is modelled and classified using four different supervised machine learning algorithms: Logistic Regression, Support Vector Machines, Decision Trees and K-Nearest Neighbours.

## Distribution of Work:

Documentation:

Chapter 1: Introduction - Sean Farrugia, Owen Agius, Francesca Mizzi

Chapter 2: Research and Literature Reviews

A Public Domain Dataset for Human Activity Using Smartphones - Owen Agius

Training Computationally Efficient Smartphone–Based Human Activity

Recognition Models – Francesca Mizzi

Energy Efficient Smartphone Based Activity Recognition Using Fixed Point Arithmetic - Sean Farrugia

Chapter 3: Implementation and Testing

Support Vector Machines - Owen Agius

Radial Kernel - Sean Farrugia

Polynomial Kernel - Owen Agius

K Nearest Neighbours - Owen Agius

Decision Trees - Francesca Mizzi

Logistic Regression - Sean Farrugia

Testing - Sean Farrugia

Chapter 4: Evaluation and Critical Analysis - Owen Agius, Sean Farrugia

Chapter 5: Conclusions - Francesca Mizzi

Implementation:

Functions for Statistical Calculations - Sean Farrugia, Owen Agius, Francesca Mizzi

Functions for reading the Datasets - Sean Farrugia

Function for Processing Raw Data - Sean Farrugia

Function for Splitting up Data - Sean Farrugia, Owen Agius, Francesca Mizzi

Data Analysis - Sean Farrugia

Modelling Data

Support Vector Machines - Sean Farrugia, Owen Agius

Logistic Regression - Sean Farrugia

K Nearest Neighbours - Sean Farrugia, Owen Agius

Decision Trees - Sean Farrugia, Owen Agius

# 2 - Research and Literature Reviews

## 2.1 - A Public Domain Dataset for Human Activity Recognition Using Smartphones

‘A Public Domain Dataset for Human Activity Recognition Using Smartphones’ [1] moves away from the traditional method of obtaining body readings and inertia sensor readings. Instead of setting up a variety of body worn sensors to collect the data, the researchers opted for a more comfortable and convenient approach.

Modern smartphones have built in gyroscopes and accelerometers. The use of smartphones with inertia sensors is an alternative approach for the Human Activity Recognition problem, as it makes the data collection procedure much more affordable, accessible and comfortable for the user. This approach also acts as a long-term solution for activity monitoring. An example of this would be a two-hour run. This activity, done using a large number of sensors attached to the body, would be very uncomfortable and impractical. Therefore, the use of smartphones nullifies this problem.

While this is an improvement when compared to the prior method of data collection, this new approach, brought forward by Anguita, Ghio, Oneto, Parra and Reyes-Ortiz, poses the issue of noisier data sets.

The researchers adapted a well-rounded approach for data collection which employs 30 volunteers ranging from ages 19 to 48. The volunteers were selected to perform 6 specific daily life tasks: standing, sitting, laying down, walking, walking upstairs and downstairs. This approach provides a good diversity of physical capabilities which allow a better-rounded classification of the activity.

The components collected via the inertia sensors range between acceleration, gravity, angular speed and magnitude, along with their respective frequencies. The readings are also computed to create a large number of statistics to assist in the classification by generalizing the readings into statistical readings.

Using a Support Vector Machine, Anguita et al. achieved a satisfying 96% accuracy in the classification of the test data. The researchers encountered challenges when classifying the stationary events: sitting, standing and laying down, resulting in a comparatively low 88% accuracy. This is due to a noticeable misclassification between the activities because of the very similar gyroscopic readings.

## 2.2 - Training Computationally Efficient Smartphone–Based Human Activity Recognition Models

Following the increasing popularity in using wearable systems in Human Activity Recognition systems (HAR Systems), Anguita et al. [2] wanted to take advantage of the rapid development of powerful processors already existing within smartphones. Companies had recently started adding sensors within smartphones such as the hybrid accelerometer and gyroscope in the latest iPhone 4. This showed researchers the benefits of adding gyroscope readings into HAR systems, observing an improvement in results, up to a 13.4% increase accuracy in classifying data.

An issue regarding implementing a HAR system using smartphones was discovered when using the K-NN classifier since it would not work in this application due to large dataset as well as the challenging computations. Thus, Anguita et al. chose to tackle two issues stemming from smartphone-based HAR systems: 1) The lack of a large gyroscope-based dataset and 2) A proper selection of useful features and effective models. The first issue was tackled through the creation of the HAR dataset, developed by the same researchers in the separate paper [1] described above, which contains a large amount of gyroscope-based data. The second issue was solved using two feature selection mechanisms as well as Support Vector Machine (SVM) models.

Since the creation of the HAR dataset was mainly described in another paper [1], not much detail is given other than that the data was collected from 30 volunteers carrying an Android smartphone around their waist performing a series of activities.

The target for the paper was to design a model which can effectively and efficiently run on the limited battery and processing power of a smartphone, which they finally chose to be a linear SVM model.

The experiments performed aimed to compare the performance of the SVM models based on linear and Gaussian kernels, with the result being that both models were substantially identical. However, they chose the linear approach since it was faster than the kernel approach.

## 2.3 - Energy Efficient Smartphone-Based Activity Recognition using Fixed-Point Arithmetic

In Bulbul [3], a prototype was applied to smartphones while being used to recognise the actions being done by the user. The main activities focused on in this paper were walking, climbing up the stairs, climbing down the stairs, sitting, standing, and laying down. Different models were tested for this paper in order to find the one with the best results. To acquire the training data for these models, 9 individuals were asked to perform the 6 activities mentioned previously while keeping their phone horizontally in a waist pocket.

During these activities, the accelerometer and gyroscope sensors in the phone were used to record the readings. A total of 6 different signals were obtained, 3 from each sensor mentioned. After that, some filtering was added to the readings of both sensors to remove any noise and increase the accuracy. Finally, a label was added at the end of each reading to describe which action was being carried out at that time. Ultimately, they collected a dataset with over 500 features. For the predictions of the actions, they tested different supervised machine learning techniques for the best accuracy.

The supervised machine learning techniques include Decision Trees, in which it was discovered that the higher the branching limit of the Trees, the more accurate the predictions. However, if they would set the branching limit too high, it would be considered as overfitting, resulting in them needing to find a lower branching limit. The next machine learning algorithm used were Support Vector Machines, which use hyper dimensional planes to distinguish between different points in space. From their testing, they concluded that the best kernel for this experiment was the polynomial kernel with a degree of 3 since it resulted in an accuracy over 99%. Finally, they used the k-Nearest Neighbours technique, a popular technique in the clustering of coefficients. The ideal value for k was found to be 3, having an accuracy of over 97%. Future improvements included potentially using more than the two sensors to achieve better results, seeing that modern phones have more sensors.

## Pros and Cons

For our task, the greatest advantage is that there is a simple way to collect data and train the chosen supervised machine learning model. The cost isn’t considered since the only hardware needed is a smartphone. Moreover, the prototype will be trained assuming that the phone will be placed in the side pocket of the user, allowing them to not be restricted by belt mount.

Should our prototype successfully become an app, many advantages can be seen. The main aim has been already established, that the app would be able to identify the action being done by the user. However, the user themselves would already know what they are doing, so a good usage of this application could be to inform other people what someone is currently doing while they are not in the same room.

It could be used to aid nurses in elderly homes to know what their patients are doing. For example, if they know that it is time for them to be asleep, but the system is still saying that the patient is currently walking, the nurse would be able to go check on the person to see what they are doing. This would be much more efficient than having to check the cameras of all the patients found inside the home.

Another great implementation to use this application for is to record your workout sessions. This can be done by our version due to the newly added actions in our model, including the activities that are more energetic. Thanks to the easy method of gathering data, more models can be easily be made to identify new activities. This could help fitness trainers make sure that their students are doing the workout that they gave them.

Although, no project is perfect it is to be expected that there are some negative aspects during this experiment, the most obvious being the accuracy of the prototype itself. Since the phone will only be capturing the data coming from the changes in accelerometer and gyroscope readings from the side pocket of the user, there is a lack of data from what the entire body is doing. For example, if an activity is being done where the upper body is mostly being used, it might be a bit difficult for the phone’s sensors to detect such an activity.

Furthermore, to properly train the chosen model with the already collected and labelled data, a lot of data must be collected. If an activity is only performed a few times while collecting supervised data for the model, it might end up being undertrained due to the lack of data fed to the model. In Anguita[1], they had to hire 30 different people to collect data for them, where in our case we only had 3. On the other hand, it is important not to overfit the data in one activity compared to the rest. If an activity has hours of data collected while another only has a few minutes, it might confuse the model with unorganised data.

# 3 - Implementation and Testing

To tackle the classification problem at hand, it was decided to attempt to classify with four different supervised machine learning algorithms. The algorithms which were adopted are:

* Support Vector Machines
* K Nearest Neighbours
* Decision Trees
* Logistic Regression

The above choices all provide a different approach for classification and therefore, ensuring a very high probability rate for correct labelling.

## 3.1 - Support Vector Machines

The objective of the support vector machine algorithm is to find a hyperplane in the Nth dimensional space which distinctly classifies the data points. Where the hyperplane can, for example: be a straight line in 2-dimensional space, and a plane in 3-dimensional space. Anything of greater dimensions would result in the hyperplane to be difficult to visualize. To separate the two or more classes of data points, there are many possible hyper-planes that could be chosen. The objective of the support vector machine algorithm is to find a plane that has the maximum margin, that is, the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

### 3.1.1 - Radial Kernel

The Radial Kernel is one of the most popular kernels used for Support Vector Machines, which uses the Radial Basis Function. This type of kernel might be comfortable to use for those types of classification models that have a large number of features, which would be impossible to draw in Euclidean Space. So what the radial basis function does is convert the represented data into an infinite coordinate by using the Taylor Series in the equation {insert equation in word}.

### 3.1.2 - Polynomial Kernel

Although the Radial Kernel is more popular in Support Vector Machine classification, the Polynomial Kernel provides a different outlook and procedure in the attempt to classify the data. This kernel does not only consider the given features of the data to determine their similarity, but also combinations of the data. Such combinations are known as interaction features. [4] An advantage this approach provides is the consideration and full expansion of the kernel prior to evaluation. Whereas, the polynomial kernel can suffer from numerical instability where the

definition usually tends to either 0 or infinity with an increasing degree. [5]

## 3.2 – K-Nearest Neighbours

The K Nearest Neighbours assumes that similar things exist in close proximity. The aim of the k-NN algorithm is to classify a new data entry given a predetermined clustered set. The new data entry is classified by comparing the distances from the other labelled variables. The distance is generally measured using Euclidean distance, but it can also be measured using either Hamming, Manhattan or Minkowski distance.

For instance, if k is equal to two, the algorithm would classify the new data entry to be in the same cluster as that of the two nearest neighbours or data points.

For the implementation of k-NN on this project, the number of neighbours is set to be seventeen. Although, most of the time the value of k is found by trial and error, generally, most implementations given an even value of classes, an odd value of k is chosen. Moreover, an odd value is generally chosen to avoid ties when classifying the new data point. If the value is set of k is set to be a relatively smaller value, such as five or seven, the result would be heavily

influenced by noise and would also result in a computationally expensive implementation . [6]

Albeit not being particularly common, a k-NN can be brought down into two variations, an eager learner or a lazy learner. An eagerly learning k-NN will construct a generalized model before performing a prediction on new given data entries. [6] Whereas, a lazy learning k-NN wait until the last minute before classifying any data point which implies that there is no requirement for learning or training of the model as all of the data points are used upon runtime. [6]

### 3.2.1 - Distance Weighted k-NN

The implementation of the k-NN consists of the parameter which sets the weight to be “distance”. When weighting on the nearest neighbours’ algorithm is used, the algorithm becomes a *global* instance. This is because all of the training set is used. It is achieved according to their distance, setting a greater weight to the closer neighbours. Although the algorithm runs slower with the weighting, it was observed that in this manner the supervised

learning algorithm yields a better accuracy. [7]

## 3.3 - Decision Trees

Decision trees are used in order to classify data by posing a number of questions related to the features within the items needed to be classified. They are built through the analysis of a training set with examples where the label/classification is already known.

Each question used to classify the data is stored within nodes and unless it is a leaf node, each internal node has two child nodes, “yes” child and “no” child. Each of these nodes together form the decision tree itself. The data which needs to be classified are filtered down each node of the tree, answering the question as they go, until they finally reach a leaf node. That data is then given the class of the leaf node it has finished in [8].

The leaf nodes within the decision tree can either be identified by a class name or in the below structure:

|  |  |
| --- | --- |
| *C1:* | *D1* |
| *C2:* | *D2* |
| *..* | *..* |
| *Cn:* | *Dn* |

In the above structure, the Ci’s are the logical conditions and the Di’s are the decision trees. Each logical condition has only attribute being

for an attribute A at a threshold T, or

for an attribute A as a value V or as a value in the subset Vi. [9]

## 3.4 - Logistic Regression

A useful algorithm which can be used in order to categorize a large amount of data into a number of different labelled classes is referred to as Logistic Regression. Despite its misleading name, Logistic Regression is actually a type of classification method rather than a regression model.

A model like Linear Regression would not be valid for this task, but a model which is able to classify the given data between different classes, which is why we considered Logistic Regression. The main difference between the two is that in Linear Regression, the model will try to fit the given data through a straight line, with the method called “least squares”. Whereas in Logistic Regression, the model will fit the data in an S-shaped graph, which has a range between 0 and 1 both included. The equation of such a graph would be like it is below:

The above equation represents the final form that the S-shaped graph would take, which is almost the same as a Sigmoid function ( ) with some additional parameters. Where in this case, does not refer to the output of the function but rather the class of the given data point since Logistic Regression works with supervised data. Another important parameter in this equation would be since without it, every logistic regression model would have the same shape. The parameter is the actual parameter that best fits the model with the training dataset. It is found by using the technique of “maximum likelihood”, which is found by using the below equation:

The likelihood is usually calculated with the help of the log of odds, where it will convert the given S-shaped curve into a straight line going from , where all the data of the bottom class are found, to , where all the data of the top class is found. The log of odds can be found with the equation where the probability is given the y-value of the S-shaped curve, which will be the new y-value of the log of odds graph. The x-values would remain the same, and in the end, you would have a straight line, similar to what you would have in Linear Regression. Through this line, the y-intercept and the gradient can be found for the straight line. Moreover, the maximum likelihood is then found by testing a number of different intercepts and gradients until the one with the largest likelihood is found.

The given data would also be labelled between two different categories. Where represents all the data in the first category and represents the remaining data of the other class. Since the output of this S-shaped curve is also going to be linear, but in the range 0 and 1, how would the classification be calculated? This is because the output of the graph is not just a real number, but the probability whether the input of the given model represents a specific class or not.

## 3.5 - Testing

After doing some research about how each of the chosen supervised machine learning algorithms work. We were ready to start testing them to see which of them would come up with the best results, and maybe even consider why they ended up having such great outcomes. However, before being able to actually start training these models, the actual training data had to be gathered and processed in order to try and replicate them the same way they did in Anguita [1], which had over 500 different features from just accelerometer and gyroscope readings.

As already stated, the application AndroSensor was used in order to obtain the readings of the sensors of our Android smartphones, where we turned on the Accelerometer, Gravity, Linear

Acceleration, and Gyroscope sensors, where each sensor is split into three values representing the X, Y and Z axes. As we thought that the more sensors we included, the better accuracy we could get from the phones readings, since we were only going to put our phone in our side pocket which would not get the best results.

Moreover, the setting of the AndroSensor application was set to output a row of data for every

0.5 seconds, so that we can later on grab 5 consecutive rows which would then represent the 2.5 seconds time window described in the papers. So for each of these 5 rows captured after each other, some statistical methods were performed on them in order to get more information out of them. Therefore, rather than just having a static reading at a particular time, one would have some statistical values of what the phone experienced during that 2.5 second time window. These statistical functions include average, median absolute deviation, standard deviation, maximum value, minimum value, signal magnitude area, energy (which is the sum of squares divided by the total number of values), interquartile range, entropy, and correlation.

All of the statistical functions mentioned will be performed on all three separate axes, except for the signal magnitude area and the correlation. This is because the signal magnitude area will be representing the normalized integral of the original values from the 3 axes at once so only one signal magnitude area will be calculated for each 3 axes. Then for correlation, there has to be some connection between two different types of data, so the correlation of each pair between the axes X, Y and Z will be calculated. Then, after all of these features were added, another column was added at the end of the row, representing which activity that row is since this is a supervised machine learning algorithm. Finally, the processed data is split into two separate datasets, where one is used to train the model and the other is used to check whether the dataset was properly trained or not.

Before starting the training of the different models chosen, we tried to visualize the processed data in order to see whether clustering is even possible between them. Since there are a large number of different features representing each row in the processed datasets, it is hard to just plot the data. Therefore, t-Distributed Stochastic Neighbouring Entities was used in order to deduct the high-dimensional data set and be able to plot it as well. This way we can see whether the data is actually clustered in some way before actually trying to train the models.

Finally, the models are trained with the gathered and processed data, with the given labels as well. The python packages for support vector machines, decision trees, k-nearest neighbours and logistic regression from the scikit-learn python package were used in order to train the models. For each model, a Grid Search was used in order to determine the best parameters to use for these models. Then the accuracy of each model and a confusion matrix of the predictions are displayed.

# 4 - Evaluation and Critical Analysis

Now comes the part that we were all waiting for, checking whether the actual prototype works or not. As already mentioned, two separate datasets were tested for this experiment, where the first one includes the usual six classes that all the other research papers that we found have included in their work. These include Standing, Sitting, Laying, Walking Upstairs and Walking Downstairs. Before starting the actual predictions, we can see how the data points look in the visualization done by the t-SNE performed in the dataset.

Already from the clustering seen in Figure #, it can be said that all the static activities performed in the training for these models, all have a distinct space and can be easily clustered. This might make sense as they are still all static, the positioning of the phone during the activity will always be different. For example, when standing the phone would have to be vertical, whilst in laying it would then have to be horizontal. So, the main differences between these three static actions must be found in the gyroscope readings of the phone.

On the other hand, for dynamic activities, there seems to be some difficulty when it comes to clustering between each other. For walking, even though close to the others, a clear cluster can be drawn, distinguishing itself from the rest of the visualization. However, for walking up and down the stairs, there seems to be quite a mash up between the two, which would make sense since they both have the exact same motion between them. Although we had assumed that with the gravity readings as well from the phone’s sensor, maybe it would have detected the difference in the height of where the person is walking.

Finally, the models were all trained and tested, where the best model turned out to be the Support Vector Machine using the Radial Basis Function kernel with an accuracy of 95%, the confusion matrix of this kernel can be found in figure #2. The parameters chosen for the RBF kernel were C=90 and the gamma value was set to ‘scale’. k-Nearest Neighbours got an accuracy of 91% with a k value of 7 and using the ball tree algorithm. Logistic Regression got an accuracy of 94.5% with L2 regularisation and One-Vs-Rest multi class option. Finally, Decision Trees got an accuracy of 92.4% accuracy with a max depth of 7.

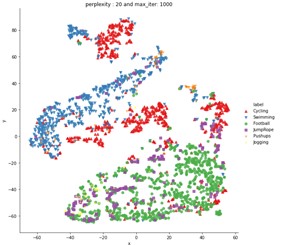


Figure 2 - Ikteb xinu

It seems that for all the models, the only class that they had difficulty in predicting was the walking upstairs classifier. However, something strange is that rather than walking downstairs, it was confused with laying down for some reason. Other than that, the rest of the classifiers were easily predicted as they are quite distinguishable between each other.

### 4.1 - Dataset 2: Six New Activities

This dataset includes six unique activities different than those of the activities used in the research papers mentioned. These activities include Cycling, Swimming, Football, Jump Rope, Push Ups and Jogging. Additionally, a Tennis dataset was also generated but was left out from this selection due to a very poor classification upon experimentation.

Considering that all of these activities entail a large emphasis on movement it would be expected that the t-SNE would find it difficult to cluster very distinctly. This can be seen in Figure 2 where although most of the Cycling data entries are segregated, a lot of outliers can be found near the Swimming cluster. Whereas, seemingly the Football data and the Jump Rope data is clustered to be very similar to each other. This can be explained due to very similar movements within each of these activities such as general jogging during football and an up and down movement during push-ups and jump-rope.

The data was collected via the Androsensor application mentioned earlier with the inertia sensors Gravity, Linear Acceleration, Gyroscope and Accelerometer only in consideration.

Finally, the activities were all trained and tested, where the best classification algorithm turned out to be the Support Vector Machine using the Radial Basis Function kernel with an accuracy of 87.3% [Fig 3] with the Logistic Regression being a close second with 87.2% [Fig 4] with L2 regularisation and One vs Rest multi class option. The other results of classification for this dataset are unfortunately not as promising as the other dataset created where the k-NN algorithm returned an 83% accuracy with a k value of 7 alongside the ball tree algorithm. Additionally, to the SVM algorithm, the Polynomial kernel returned an 84% accuracy with the same parameters of the RBF being a c value of 90 and a scale gamma variable. Finally, the

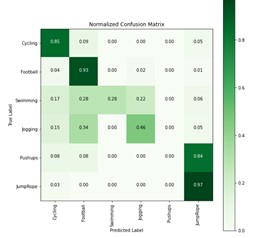
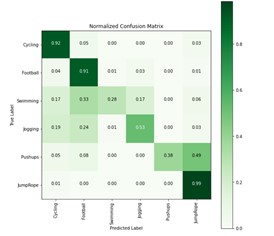
Decision Trees algorithm classified with 83.5% accuracy considering a maximum depth of 7.

Figure 4 – zobbi

Figure 3 – liba

Considering the confusion matrices found in Figures 3-7 hints, difficulty in proper classification of the Swimming, Jogging and Push Ups activities. This can be understood and explained due to the heavy number of outliers found in the Swimming cluster and the Jogging and Pushups activities finding home in and very close to the Football data entries.

# 5 - Conclusions

# References

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “A Public Domain Dataset for Human Activity Recognition Using Smartphones,” *European Symposium on Artificial Neural Networks*, vol. 3, Apr. 2013, [Online]. Available: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2013-84.pdf.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Training Computationally Efficient Smartphone–Based Human Activity Recognition Models,” in *International Conference on Artificial Neural Networks*, Sep. 2013, pp. 426–433, [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-40728-4\_54.

[3] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Energy Efficient Smartphone-Based Activity Recognition Using Fixed-Point Arithmetic,” *Journal of Universal Computer Science*, vol. 19, no. 9, pp. 1295–1314, May 2013, [Online]. Available: https://upcommons.upc.edu/bitstream/handle/2117/20437/jucs\_19\_09\_1295\_1314\_anguita.pdf;sequence=1.

[4] Y. Goldberg and M. Elhadad, “splitSVM: Fast, Space-Efficient, non-Heuristic, Polynomial Kernel Computation for NLP Applications,” *Proceedings of ACL-08: HLT, Short Papers*, pp. 237–240, Jun. 2008.

[5] C.-J. Lin, “Machine learning software: design and practical use,” *Machine Learning Summer School*, 2012. <https://www.csie.ntu.edu.tw/~cjlin/talks/mlss_kyoto.pdf>.

[6] Avinash Navlani, “KNN Classification using Scikit-learn,” *DataCamp Community*, 2018. https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

‌[7]“Distance weighted k-NN algorithm,” *Data-machine.net*, 2021. http://www.data-machine.net/nmtutorial/distanceweightedknnalgorithm.htm

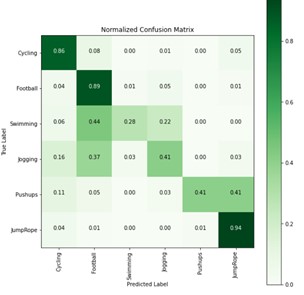
[8] C. Kingsford and S. L. Salzberg, “What are decision trees?,” *Nature Biotechnology*, vol. 26, no. 9, pp. 1011–1013, Sep. 2008, doi: 10.1038/nbt0908-1011.

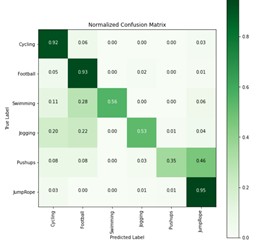
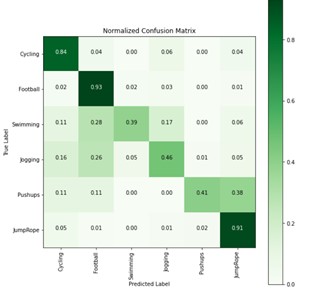
‌[9] J. R. Quinlan, “Simplifying Decision Trees,” *International journal of man-machine studies*, vol. 27, no. 3, pp. 221–234, Sep. 1987, [Online]. Available: https://dspace.mit.edu/bitstream/handle/1721.1/6453/aim-930.pdf?sequence=2.

Figure 2 - Owen

Figure 3 - Owen

Figure 4 - Owen

Figure 5 - Owen

Figure 6 - Owen Figure 7 – Owen