|  |  |
| --- | --- |
| **A study on Heterogeneity Activity Recognition Data Set** | Chandra Shakhar Kundu  CS 555: Data Science |

Contents

[a. Background 2](#_Toc7468400)

[b. Objective 3](#_Toc7468401)

[c. Methodology 3](#_Toc7468402)

[c.1. Pre-Processing 4](#_Toc7468403)

[c.2. Model Analysis 5](#_Toc7468404)

[c.2.1. K-Nearest Neighbors 5](#_Toc7468405)

[c.2.2. Support Vector Machine 6](#_Toc7468406)

[c.2.3. Decision Tree 7](#_Toc7468407)

[c.2.4. Random Forest 8](#_Toc7468408)

[d. Result analysis: 9](#_Toc7468409)

[e. Discussion 12](#_Toc7468410)

[f. Conclusion 12](#_Toc7468411)

[g. Reference 12](#_Toc7468412)

# Background

Smart devices such as mobile phones and smart watches and their increasing range of capabilities, especially in terms of wireless connectivity and sensors, has enabled a wide spectrum of applications that are creating our daily life more flexible. These devices equipped with GPSs, accelerometers, gyro meters, and many different sensors, are now able to gather contextual information such as location and motion patterns. Furthermore, additional sensors can be easily integrated into these devices to acquire physiological data, e.g., body temperature, heart rate, respiration rate, etc. allowing for real-time health monitoring.

Among the sensors available on smart device platforms, the accelerometer is one of the earliest and most ubiquitous. This sensor can measure acceleration, the rate of change of velocity, and detect changes in orientation of the device. It computes the linear acceleration of the device on the X- axis (lateral), Y-axis (longitudinal), and Z-axis (vertical). This linear acceleration pattern can be used to recognize human activity. For example, if a user changes his/ her activity from walking to biking, there will be abrupt change of acceleration pattern along the vertical axis.

Another sensor is used in HAR is the gyroscope sensor that can also provide information about device’s orientation, but with greater precision. This sensor has been used in smart devices to measure the device’s rotation rate by detecting the roll, pitch, and yaw motions of them along the x, y, and z axis respectively.

|  |  |  |
| --- | --- | --- |
| Sensor | Common Use | Description |
| Accelerometer | Motion Detection | Detect acceleration along x, y and z axis |
| Gyroscope | Rotation Detection | Detect rate of rotation along x, y and z axis |

For our project, we use the heterogeneity dataset for Human Activity Recognition which is taken from University of California Irvine (UCI) machine learning data repository. This data is gathered by conducting an extensive analysis on 9 users using 31 smartphones, 4 smartwatches and 1 tablet, representing 13 different models from 4 manufacturers, running variants of Android and iOS, respectively. The dataset contains the readings of two motion sensors commonly found in smartphones. Reading were recorded while users executed activities scripted in no specific order carrying smartwatches and smartphones.

Activities: ‘Biking’, ‘Sitting’, ‘Standing’, ‘Walking’, ‘Stair Up’ and ‘Stair down’.  
Sensors: Sensors: Two embedded sensors, i.e., Accelerometer and Gyroscope, sampled at the highest frequency the respective device allows.  
Devices: 4 smartwatches (2 LG watches, 2 Samsung Galaxy Gears)  
8 smartphones (2 Samsung Galaxy S3 mini, 2 Samsung Galaxy S3, 2 LG Nexus 4, 2 Samsung Galaxy S+)  
Recordings: 9 users

The activity recognition environment and scenario has been designed to generate many activity primitives, yet in a realistic manner. Users took 2 different routes for the biking and walking, and 2 different set of stairs were used for the stairs up and down.

The csv files contain all smartphone accelerometer samples from all devices and users. The csv files consist of the following columns:   
'Index', 'Arrival\_Time', 'Creation\_Time', 'x', 'y', 'z', 'User', 'Model', 'Device', 'gt'  
The null class is defined as null in the gt (groundtruth) column, whereas the rest of the classes can be seen in the column.

# Objective

The widespread availability of wearable sensors such as smart phones, smart watches, etc. in everyday lives is generating an ample amount data-set through which we can be used to predict human activity. In this project, we used the heterogeneity activity recognition data set collected from different smartphones and smart watches equipped with accelerometer and gyroscope sensors, to predict human activities such as biking, sitting or standing and walking. There are many statistical and machine learning classification algorithm which can be used to predict human activities. A comparative study among these algorithms can help us to find the most suitable one that should be used in human activity recognition. In this project, first, we implement svm, decision tree classifier and random forest to this data set and then a comparative study on scores along with their ROC is discussed.

# Methodology

From our dataset and problem, since we have to predict human activity based on dataset features, we understand it is a classification problem. Here, we followed the below steps which include preprocessing of data, model selection and training the data using and finally, evaluation of the models.

## c.1. Pre-Processing

First, we merge all the dataset into one dataframe. Now we try to find to detect all the missing and duplicate value in the data. Luckily, the data is almost preprocessed except some missing values in the ‘gt’ column. Since “gt” is our class attribute, so we drop all the row where “gt” has null value. Our data has some extra features such as 'Index', 'Arrival\_Time', 'Creation\_Time', 'User', 'Model', 'Device', for our project, which are not necessary. Because when we predict human activity, we only use the sensor type and their measurement along the three axes. We drop those extra features and add an extra feature called ‘sensor’ which contains the types of sensors is used to collect the data.

Moreover, our data has six unique classes: ‘Biking’, ‘Sitting’, ‘Standing’, ‘Walking’, ‘Stair Up’ and ‘Stair down’. We factorize the feature: sensors and the class: activity into integers to handle those easily. Then we determine the correlation among the features and find no significant correlation among them.

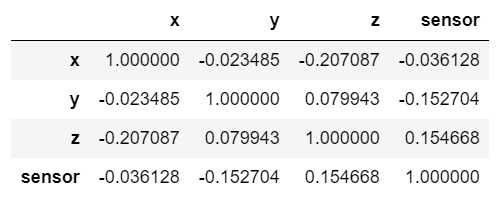


Table: Correlation among features

Now we do cluster analysis to see any pattern in our data. For our hardware limitation, instead of taking the whole data, we take a random sample of 100000 row. Now, our data has four features: x, y, z and sensor, we reduce it into two feature using principal components to draw a scatter plot. Principal component analysis (PCA) is a popular dimensionality reduction technique within the field of machine learning and statistics. PCA archives data compression by projecting data onto a linear orthogonal subspace such that the variance in the projected data is maximized.

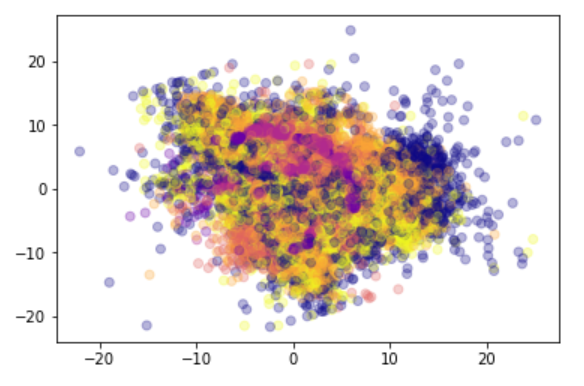
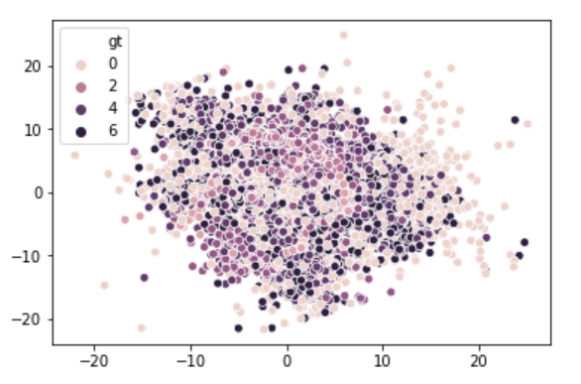


Figure 1: Scatter Plot For 6 classes after PCA

From the scatter plot, we see no clear pattern in the data and more over from the Figure 1(right) that is drawn using seaborn, PCA only recognize 4 classes. We know that to keep the overall cost low, mobile devices are often equipped with low cost sensors, which are often poorly calibrated, inaccurate, and of limited granularity and range, compared to dedicated sensors for HAR. In our dataset, there are 6 classes. Poorly calibrated sensor may not able to differentiate between sitting and standing or walking and stair up and down. So, we combine sitting and standing together, and walking, stair up and down and biking together. Now we have two classes and draw scatter plot for them.

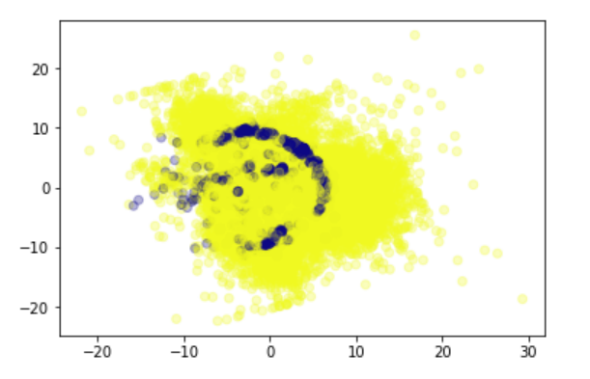


Figure 2: Scatter Plot For 2 classes after PCA

This time data has clearer pattern than before. Now we can proceed with this data to do further analysis.

## c.2. Model Analysis

### c.2.1. K-Nearest Neighbors

KNN algorithm is one of the simplest classification algorithms and it is one of the most used learning algorithms. KNN can be used for classification — the output is a class membership (predicts a class — a discrete value). An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.

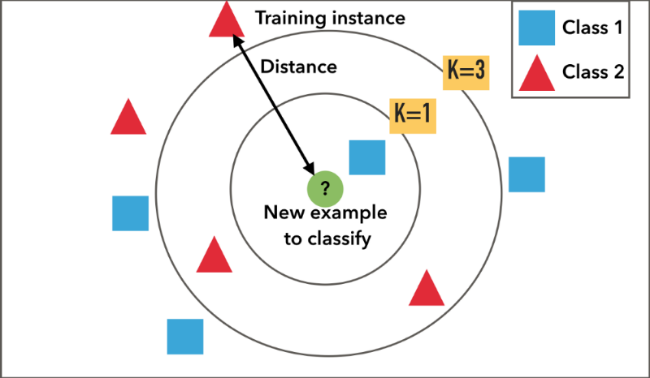


Figure 3: K Nearest Neighbors

**Justification:**   
KNN algorithm is very simple to understand and equally easy to implement. To classify the new data point KNN algorithm reads through whole dataset to find out K nearest neighbors. Given it’s an instance-based learning; KNN is a memory-based approach. The classifier immediately adapts as we collect new training data. It allows the algorithm to respond quickly to changes in the input during real-time use. For HAR, we need an algorithm that respond quickly to changes in real-time data. So KNN is a good option for this dataset.

### c.2.2. Support Vector Machine

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outlier detection. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

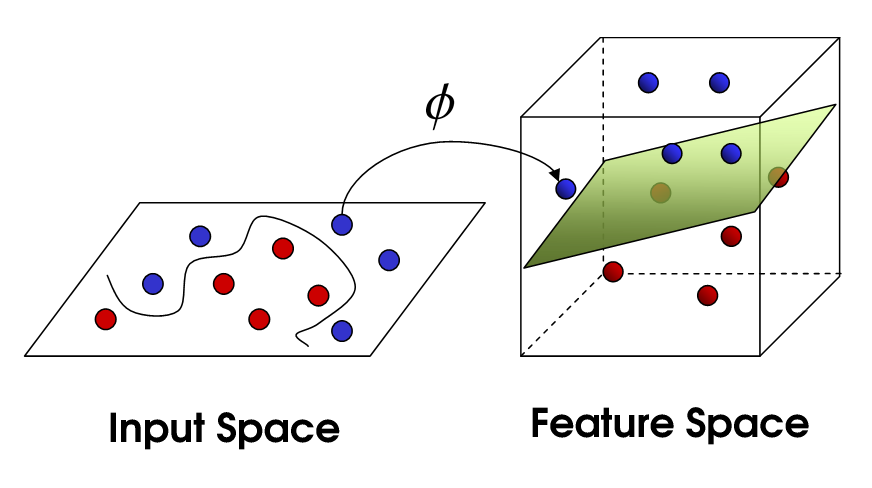


Figure 4: Support Vector Machine

**Justification:**  
Compared to logistic regression SVM helps us to give more optimal boundary using maximum margin, which will lead to better performance. Classifier in SVM depends only on a subset of points. Since we need to maximize distance between closest points of two classes, we need to care about only a subset of points unlike logistic regression.

c.2.3. Decision Tree  
The decision tree classifiers organized a series of test questions and conditions in a tree structure. The following figure shows a example decision tree for predicting whether the person cheats. In the decision tree, the root and internal nodes contain attribute test conditions to separate records that have different characteristics. All the terminal node is assigned a class label Yes or No.

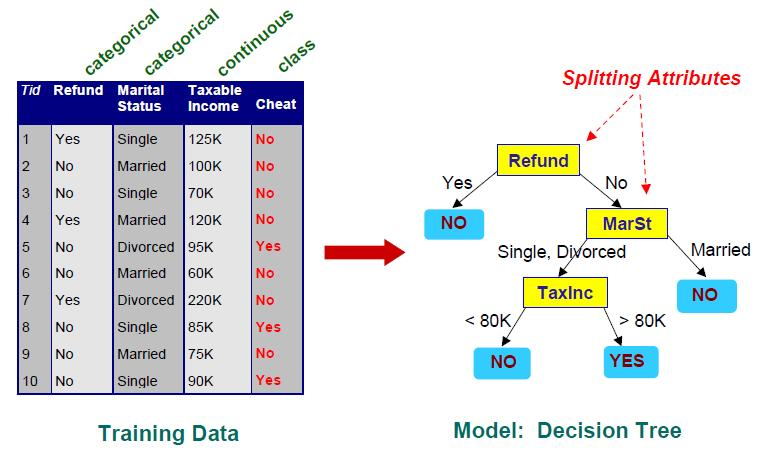


Figure 5: Decision Tree

**Justification:**   
Decision trees require relatively little effort from users for data preparation. As from the cluster analysis, our data set is highly nonlinear. This nonlinear relationships between variables will result in failing checks for simple regression models and thus make such models invalid. Moreover, we converted our problem into binary class problem where decision tree has the advantage to predict the class without high computational efforts.

### c.2.4. Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

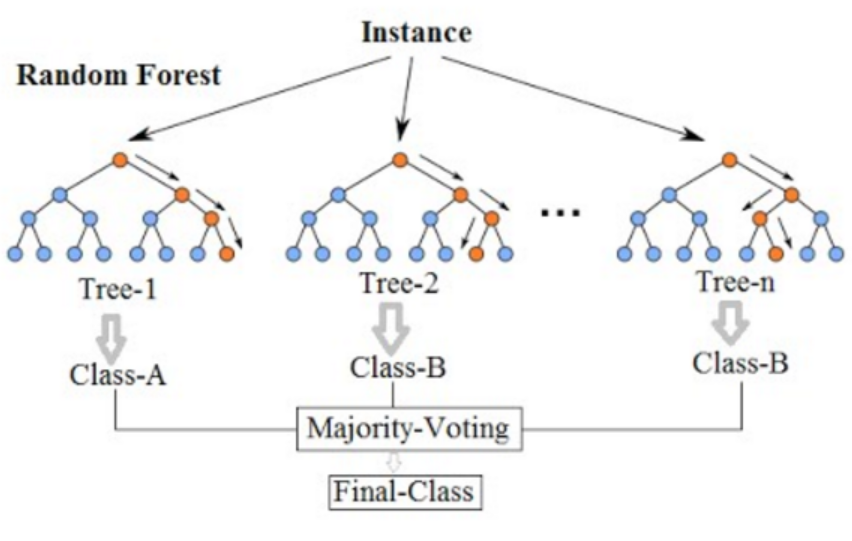


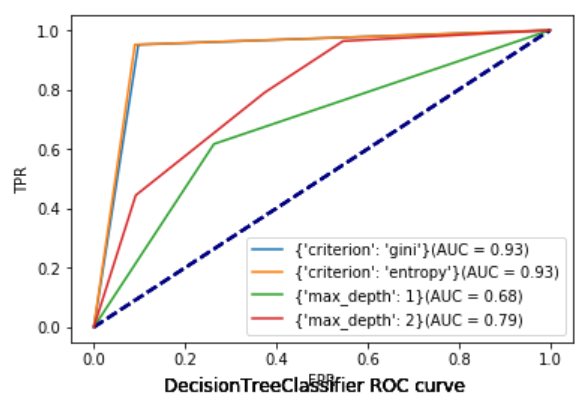
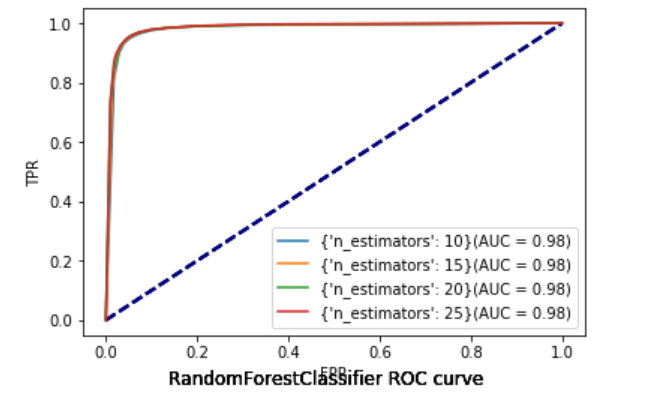
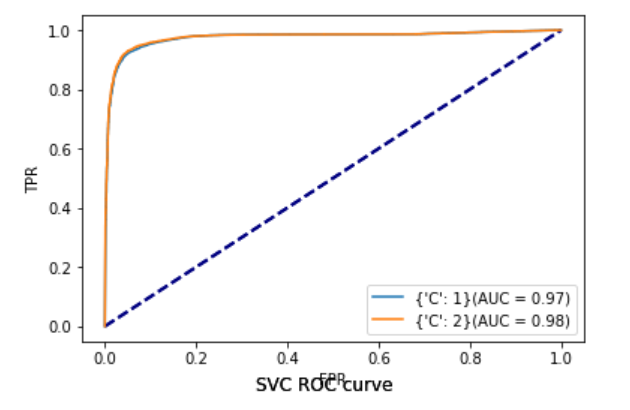
Figure 6: Random Forest

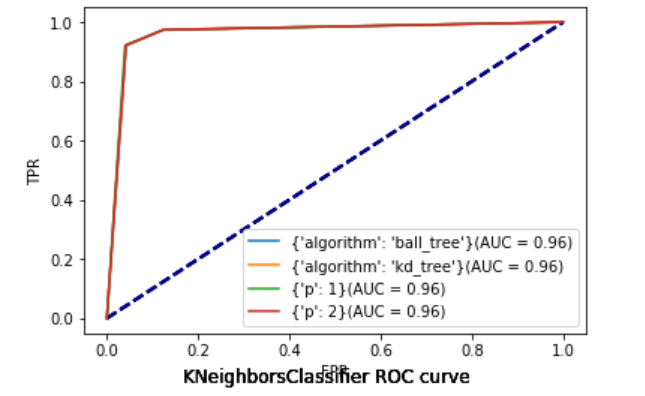
**Justification:**

Random forests are extremely flexible and have very high accuracy. They also do not require preparation of the input data. The computational cost of training a random forest is quite low. Since our data is very large to train in a personal computer, so random forest is a good option to solve this hardware limitation.

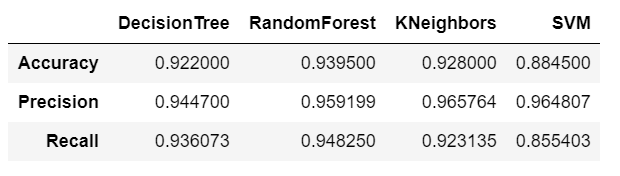
# Result analysis:

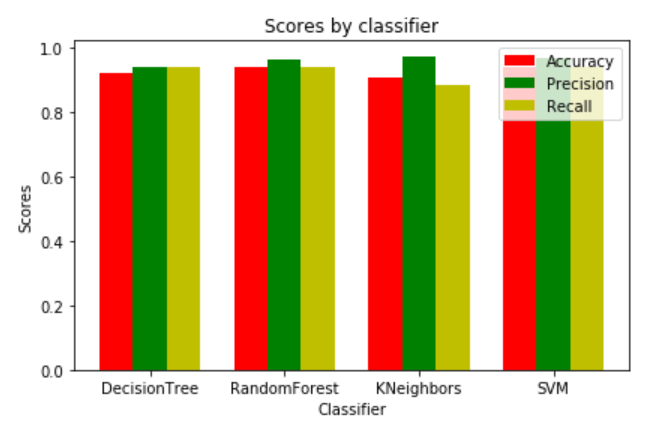
To evaluate the performance of K-Nearest Neighbors (KNN), Decision Tree, Random forest and Support Vector Machine (SVM) on our dataset, we used different parameter for each of the algorithm. The ROC for each algorithm using different parameter given below:



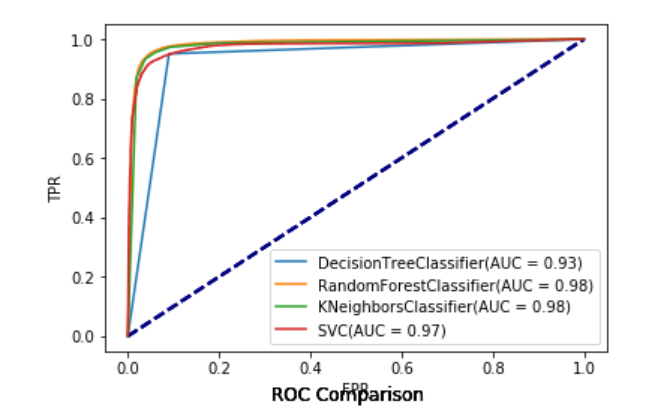


From ROC, all the four classifier is pretty steady with different parameters except the decision tree. Now we take the best parameters for each of the classifiers for which we get maximum AUC and compare with each other.





We can also draw ROC for all the four classifiers. This will give us idea about the performance of each classifier compare to others.



So far, we compared all the four model in terms of accuracy, precision, recall and ROC. Among the four model, in terms of ROC, Random forest and KNN outperform other two classifier. In terms accuracy and recall, Random forest is the best classifier. If you take account of precision, KNN performs better than other three. Since, Random Forest is very computationally efficient and it has steady and best performs in terms of several scores, I suggest using this classifier for our dataset.

# Discussion

In result analysis, we compared all the four models and suggested Random forest would be best classifier for this dataset. Furthermore, I want to give some insights about preprocessing and modeling our dataset. As our dataset is very noisy, I choose only few features to modeling data and converted multiclass problem into two class problem. However, for this dataset, to analysis multiclass problem, we need more features for instance, creation\_time and activation\_time given in the dataset can be used. In that case, some time-series related model would perform better than these four classifiers. Moreover, combination of different models may also provide better result.

# Conclusion

The recent development of technology is making every device in our world smart whenever possible. One of the goals of making smart systems is to recognize daily human activities by using a smart phone or a smart watch to encourage a safe, healthy life for people concerned about their health. In this report, we have discussed the background of our project. We have done a cluster analysis using PCA, preprocessed the data, presented different classification technique and then apply the techniques to recognize human activities. We have done an extensive result analysis in terms of accuracy, precision, recall and ROC. We have also discussed about further techniques that can be used in human activity recognition.

# Reference

1. *Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen "Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition" In Proc. 13th ACM Conference on Embedded Networked Sensor Systems (SenSys 2015), Seoul, Korea, 2015. [*[*Web Link*](http://dx.doi.org/10.1145/2809695.2809718)*]*
2. *Dataset URL:* [*http://archive.ics.uci.edu/ml/datasets/Heterogeneity+Activity+Recognition*](http://archive.ics.uci.edu/ml/datasets/Heterogeneity+Activity+Recognition)