**Wearable computing Report 2017**

**Accelerometer’s Data Classification of Body Postures and Movements**

Submitted By:

**Anjali Barodia**

**Batch 30 - INSOFE, Bengaluru**

****

### **International School of Engineering, Bengaluru**

### **10th November, 2017**

**Contents**

**Abstract -------- ------------------------------------------------------------------------------------------------------- 2**

**1. Introduction -------------------------------------------------------------------------------------------------------- 2**

1.1 Problem Statement -------------------------------------=-------------------------------------------------------- 3

1.2 Data Description ------------------------------------------------------------------------------------------------- 3

**2. Exploratory Data Analysis -------------------------------------------------------------------------------------- 4**

2.1 Data Cleansing --------------------------------------------------------------------------------------------------- 4

2.2 Handle Missing Data -------------------------------------------------------------------------------------------- 5

2.3 Outlier Detection ------------------------------------------------------------------------------------------------ 5

2.4 Data Standardization -------------------------------------------------------------------------------------------- 6

2.5 Correlation Plot -------------------------------------------------------------------------------------------------- 7

2.6 Feature Selection ------------------------------------------------------------------------------------------------ 8

2.7 Data splitting ----------------------------------------------------------------------------------------------------- 8

**3. Models Building and Evaluation -------------------------------------------------------------------------------- 9**

3.1 Multi-nominal Logistic Regression ---------------------------------------------------------------------------- 9

3.2 stepAIC Model --------------------------------------------------------------------------------------------------- 11

3.3 Decision Tress (C5.0) ------------------------------------------------- ----------- ----------------------------- 11

3.4 SVM --------------------------------------------------------------------------------------------------------------- 13

3.5 Naive Bayes Model ---------------------------------------------------------------------------------------------- 15

3.6 Knn Model ------------------------------------------------------------------------------------------------------ 15

3.7 Random Forest Model ------------------------------------------------------------------------------------------ 16

**Conclusion and Future Work -------------------------------------------------------------------------------------- 17**

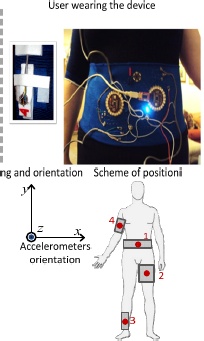
**Reference --------------------------------------------------------------------------------------------------- ----------- 18**

**Abstract**

The purpose of this internship is to work on Wearable Computing Data-set of Human Activity Recognition (HAR). HAR has reported on systems showing good overall recognition performance. As a consequence, HAR has been considered as a potential technology for e-health systems.

Our goal is to purpose a machine learning based HAR classifier. They provided a full experimental description that contains the HAR wearable devices setup and a public domain data-set comprising 165,633 samples. They consider 5 activity classes, gathered from 4 subjects wearing accelerators mounted on their waist, left thigh, right arm, and right ankle. As basic input features to our classifier we use 12 attributes derived from a time window of 150ms. Our task is to analysis the sensors data to so as to recognize classes accurately : standing, sitting, sitting-down, standing-up and walking.

**1. Introduction**

****

With the development of new technologies, life

expectancy also get increased.So,to enable a more

independent and safer life to the elderly and the

chronically ill has become a challenge.

Human Activity Recognition(HAR) is an active

research area, results of which have the potential

to benefit the development of assertive techno

-logies in order to support care of the elderly,

the chronically ill and people with special needs.

Activity recognition can be used to provide infor

-mation about patients’ routines to support the development of e-health systems. Two approaches are commonly used for HAR: image processing and use of wearable sensors.

The image processing approach does not require the use of equipment in the user’s body, but imposes some limitations such as restricting operation to the indoor environments, requiring camera installation in all the rooms, lighting and image quality concerns and mainly, users’ privacy. The use of wearable sensors minimizes

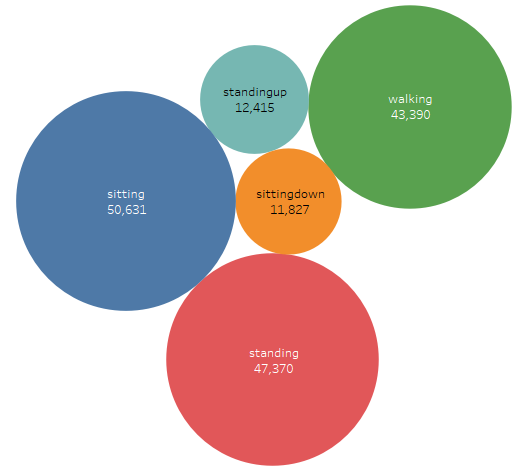
these problems, but requires the user to wear the equipment through extended periods of time. Hence, the use of wearable sensors may lead to inconveniences with battery charges, positioning calibration of sensors.

It was further observed that few works provide public Data-sets for bench-marking and there is no sufficient information on the specification and orientation of the sensors deployed. In view of such results, it was decided for the investigation of

activities recognition by means of wearable accelerators approach.

In this project we use of 4 accelerator meters reading positioned in the waist, thigh, ankle and arm. We

have data from 4 people in different static postures and dynamic movements.

**1.1 Problem Statement**

An experiment conducted on 4 healthy people on

a time-span of 8 hours, sensor data is collected

on for each person for 2 hours. Our task is to

classify their body posture and movements.

So , it’s a multi-nominal classification problem.

**1.2 Data Description**

Data-set collected data during 8 hours of activities, 2 hours with each one of the 4 subjects: 2 men and 2 women, all adults and healthy. The protocol was to perform each activity separately. Total instance are 165,633 and profile is diverse: women, men, young adult and one elder.

**Attribute Information:**

user (text) - string

gender (text) - factor(woman/man)

age (integer) -

how\_tall\_in\_meters (real) -height(cm)- convert into numeric

weight (int) - kg

body\_mass\_index (real)

x1 (type int, contains the read value of the axis 'x' of the 1st accelerometer, mounted on waist)

y1 (type int, contains the read value of the axis 'y' of the 1st accelerometer, mounted on waist)

z1 (type int, contains the read value of the axis 'z' of the 1st accelerometer, mounted on waist)

x2 (type int, contains the read value of the axis 'x' of the 2nd accelerometer, mounted on the left thigh)

y2 (type int, contains the read value of the axis 'y' of the 2nd accelerometer, mounted on the left thigh)

z2 (type int, contains the read value of the axis 'z' of the 2nd accelerometer, mounted on the left thigh)

x3 (type int, contains the read value of the axis 'x' of the 3rd accelerometer, mounted on the right ankle)

y3 (type int, contains the read value of the axis 'y' of the 3rd accelerometer, mounted on the right ankle)

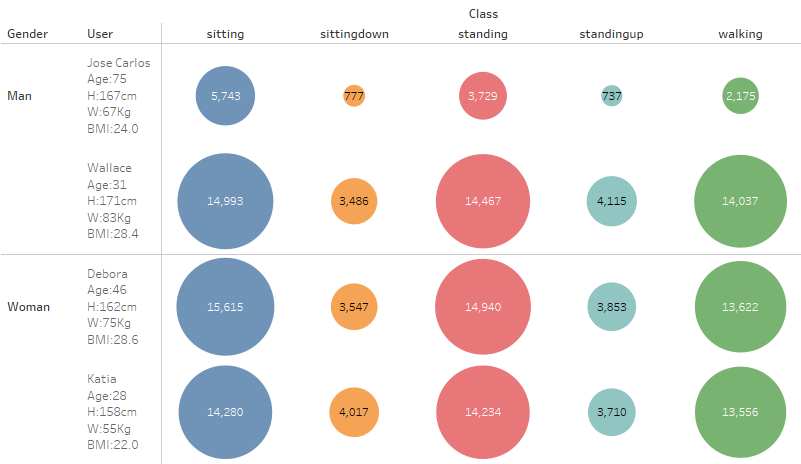
z3 (type int, contains the read value of the axis 'z' of the 3rd accelerometer, mounted on the right ankle)

x4 (type int, contains the read value of the axis 'x' of the 4th accelerometer, mounted on the right upper-arm)

y4 (type int, contains the read value of the axis 'y' of the 4th accelerometer, mounted on the right upper-arm)

z4 (type int, contains the read value of the axis 'z' of the 4th accelerometer, mounted on the right upper-arm)

The distribution of the users with classes is illustrated here :



**2 .Exploratory Data Analysis**

**2.1 Data Cleansing**

**Correct the data types:** By seeing the data structure we found that some of the data-types are not in correct format. So, we use type conversion to correct the data types.

Following is the corrected data types result we obtained :

**chr** : user

**Factor** : gender, class

**int** : age, weight,

x1, y1, z1,

x2, y2, z2,

x3, y3, z3,

x4, y4, z4

**num** : height, bmi

**Features / Dimensions:** user, gender, age, weight, height, bmi, x1, y1, z1,x2, y2, z2, x3, y3, z3, x4, y4, z4

**Target :** class

5 ordinal levels: sittingdown, sitting, standing, standingup, walking

**2.2 Handle Missing Data**

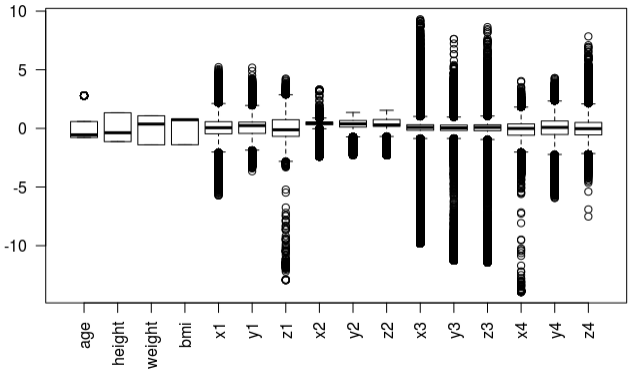
There is only 1 null values, so just use knnImputation function with k=400 and imputed the null value.

We use k = 400 because as there around 165,633 data. K should be near to sqrt(no of training samples).

**2.3 Outlier Detection**

Box Plot: A box plot is a graphical rendition of statistical data based on the minimum, first quartile, median, third quartile, and maximum.The top of the rectangle indicates the third quartile, a horizontal line near the middle of the rectangle indicates the median, and the bottom of the rectangle indicates the first quartile. A vertical line extends from the top of the rectangle to indicate the maximum value, and another vertical line extends from the bottom of the rectangle to indicate the minimum value. Bubbles in box-plot show the outliers.

**Box plot of features:**



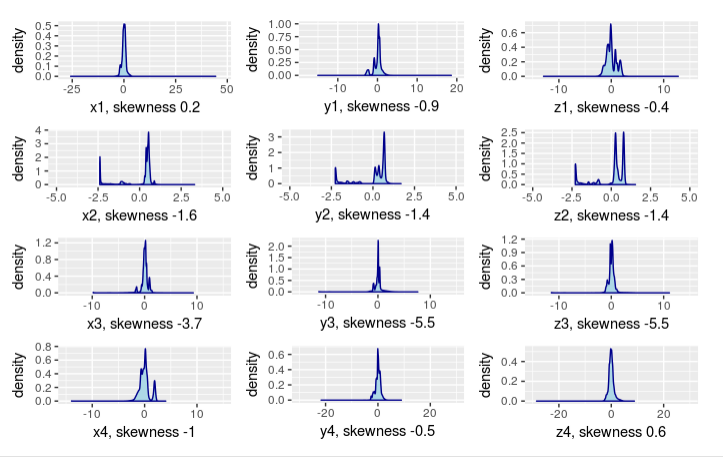
From the box pot we see there are so many extreme points are present. That does not means all are not bad, that because human activities are really complex and diverse. Thus this ext rem points of sensor data may be possible.

Even though we see from the further analyses we found there are some points which are really rare and belongs to one particular class so we use **Winsorizing** the data.

Winsorizing or winsorization is the transformation of statistics by limiting extreme values in the statistical data to reduce the effect of possibly spurious outliers.

Thus we removed around 30 records.

Without performing any standardization we see the distribution of data as below:



We see from the graph distribution of each numerical features and with there skewness receptively.

For accelerometer1, 3, and 4 we found the skewness is fine but for accelerometer 2 the data x2,y2 and z2 are highly negative skewed. So, we will try some transformation and see how transformation making an impact on skewness.

**2.4 Data Standardization**

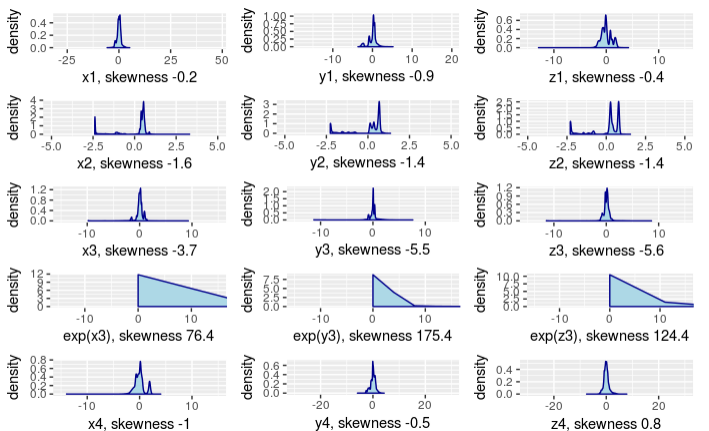
**Standardize the data:** Standardization the data to bring down all features on the same scale and see weather there is any change the distribution.

We notice there is no change in distribution. It is same. But standardizing the data is always help in model building.

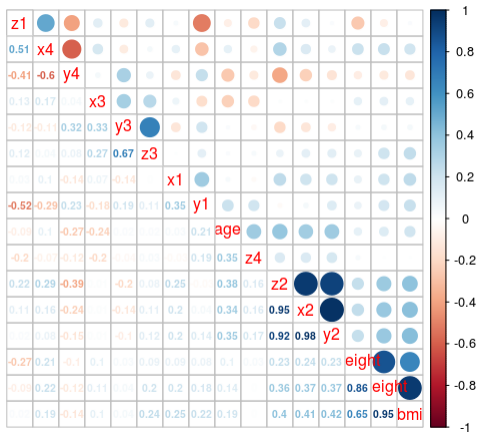
After standardizing the data we performed some transformation and try to see the how skewness is varying.

For x2, y2 and z2 we see there is negative skewness, so try exponential transformation and we found that skewness after transformation is remains same.

Density Distribution of scaled-features and accelerometer2’ s exponential transformation result:



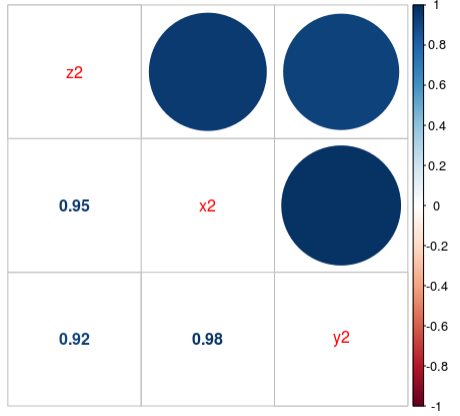
**2.5 Correlation Plot**

****Correlation plot t: this plots showing correlations among pairs of variables in X. Through this we try to explore the relationship (linear association) among features.

All correlations have two properties: **strength** and **direction**.

* The strength of a correlation is determined by its numerical value.
* The direction of the correlation is determined by whether the correlation is positive or negative.

* Positive correlation: Both variables move in the same direction. As one variable decreases, the other variable also decreases.
* Negative correlation: The variables move in opposite directions. As one variable increases, the other variable decreases.

**Fig: Correlation plot (x2, y2, z2)**

We found that the correlation among x2, y2 and z2 is very high. That can seen in the figure:

So we can drop any one of these feature and see in model, how its accuracy and error are affecting. This will help in feature selection.

**2.6 Feature Selection**

Our problem is to classify the body movements and posture correctly and purpose machine learning model which able to do this task whenever new data comes. We have 19 features including target class. Broadly we have accelerometers data and user data as a independent features.

From the analysis we found that to classify the classes accelerometers data is sufficient to perform the task,

so user data like his name, height, weight, age and bmi is not much useful in the prediction.

At the time model building all this analysis will further help us to drop features and select important features.

This below points should consider at the time of model building:

* Correlation plot : from x2, y2 and z2 we have to see which feature to select, as these features are highly correlated.
* Important features from decision tree.- This will help us to know which accelerometers data is most important to recognize the human activities.

**2.7 Data splitting**

Split the data into train and test. For this use stratified sampling which take care the proportions. As there are 5 classes and there proportion is as follows:

* sitting – 31%
* sittingdown – 7%
* standing – 0.29%
* standingup – 7%
* walking – 26%

As we see the proportions for sittingdown and standingup is 7% only, so to make the ratio same in training and test data we are using createDataPartition function of caret package.

Here the split ratio from train set to test set is 7:3 and createDataParttion function used to create the balanced splits of the data.

* **T**rain data split result:

**Dim. of Train Data: 124228 19**

sitting sittingdown standing standingup walking

0.31 0.07 0.29 0.07 0.26

* Test data split result:

**Dim. of Test Data: 41405 19**

sitting sittingdown standing standingup walking

0.31 0.07 0.29 0.07 0.26

To train a model we are going to use train data set and to predict on unseen data we will use test data set.

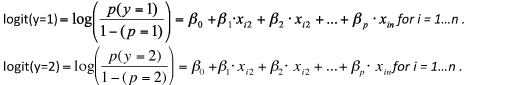
**3. Models Building and Evaluation**

**3.1 Multi-nominal Logistic Regression:**

Multinomial Logistic Regression is the linear regression analysis to conduct when the dependent variable is nominal with more than two levels. Thus it is an extension of logistic regression, which analyzes dichotomous (binary) dependents.

Multinomial regression is a multi-equation model, similar to multiple linear regression.  For a nominal dependent variable with k categories the multinomial regression model estimates k-1 logit equations.

For example for 3 classes it will generate 2 equation:



We have build multiple models on different set of data and found that multi-nominal model on whole data i.e user data and all 4 accelerometers it give accuracy 83.32% on train and 83.25% on test. As, we have to recognize the activities correctly, user’s data like name, age, weight, height and gender doesn’t help in recognize accurately, it may help use to see in which class they fall maximum times but at a moment whats their activity we won’t able to recognize, that can be done by sensor data only.

So we build Model 2 on sensor data only and found that accuracy decreased by 0.67 only. That means user data is not helping us in classifying the classes. We can move on by considering sensor data only.

While pre-processing we saw that accelerometer - 2 data (placed on left thigh), x2, y2 and z2 are highly correlated so we drop 2 columns one by one and will see which feature is more important. As we see in the table we found that using any one the features among 3 features gives the same accuracy. And among all 3 feature z2 give a very slightly good accuracy over other two.

Next we see how well each accelerometer data individual is helping us classifying the activities. We found that with using one accelerometer does not help us in classifying data correctly. So, we tried with 3 accelerometer’s data (waist, thigh and ankle).It also give fair result i.e. Accuracy is 75.2% 1 on train and 74.61% on test data.

Then we building model using each acceleometer and found that which accelerometer contributing in recognizing which classes :

Accelerometer 1 (Placed on waist) – Help in recognizing standing, sitting and walking classes

Accelerometer 2 (Placed on thigh) – Help in recognizing standing, sitting and walking classes

Accelerometer 3 (Placed on ankle) – Help in recognizing standing, sitting and walking, standingup classes

Accelerometer 4 (Placed on arm) – Help in recognizing sittingdown classes

This above observation is derived from the table : Multinominal Models Results

**Table: Multinominal Models Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | **Specificity (%)** | | | | | | **Remarks** |
| **Multinominal Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |  | |
| **Model 1** | **Train** | **83.32** | 97.91 | 73.98 | 76.08 | 72.96 | 79.07 | 99.78 | 97.04 | 93.06 | 96.75 | 92.02 | All data (user and sensor data) | |
| **Test** | **83.25** | 97.97 | 74.01 | 75.84 | 73 | 78.99 | 99.8 | 96.98 | 92.98 | 96.74 | 92.06 |
| **Model 2** | **Train** | **82.68** | 97.70 | 71.13 | 72.88 | 72.84 | 82.68 | 99.75 | 96.85 | 94.92 | 96.71 | 89.99 | Only on sensor data- all 4 accelerometer data | |
| **Test** | **82.71** | 97.76 | 71.6 | 72.59 | 73.38 | 82.98 | 99.77 | 99.74 | 94.95 | 96.7 | 89.95 |
| **Model 3** | **Train** | **80.72** | 97.20 | 64.74 | 68.75 | 65.43 | 85.12 | 99.48 | 96.07 | 95.65 | 95.32 | 89.78 | On sensor data except x2,y2 | |
| **Test** | **80.50** | 97.14 | 64.84 | 68.29 | 64.34 | 85.25 | 99.56 | 96.06 | 95.65 | 95.21 | 89.6 |
| **Model 4** | **Train** | **80.02** | 96.10 | 62.73 | 69.09 | 61.15 | 84.6 | 98.83 | 96.09 | 95.5 | 95.26 | 89.64 | On sensor data except x2,z2 | |
| **Test** | **79.91** | 95.95 | 62.57 | 68.78 | 61.1 | 84.85 | 98.85 | 96.04 | 95.48 | 95.52 | 89.63 |
| **Model 5** | **Train** | **80.02** | 96.41 | 63.41 | 68.58 | 61.92 | 84.65 | 98.80 | 96.08 | 95.39 | 95.23 | 89.78 | On sensor data except y2,z2 | |
| **Test** | **79.81** | 96.18 | 63.51 | 68.25 | 61.19 | 84.7 | 98.80 | 96.05 | 95.38 | 95.14 | 89.68 |
| **Model 6** | **Train** | **60.19** | 89.79 | 20.11 | 47.68 | 30.4 | 49.14 | 94.86 | 92.02 | 86 | 92 | 81.1 | On sensor 1 only | |
| **Test** | **60.37** | 90.05 | 13.7 | 47.85 | 29.87 | 49.12 | 95.10 | 92.8 | 89.7 | 92.81 | 80.92 |
| **Model 7** | **Train** | **67.67** | 81.14 | 29.41 | 60.77 | 24.58 | 62.73 | 96.82 | 92.6 | 97.34 | 92.79 | 82.1 | On sensor 2 only | |
| **Test** | **67.62** | 80.94 | 12 | 60.66 | 26.99 | 63.05 | 96.82 | 92.8 | 97.28 | 92.87 | 82.06 |
| **Model 8** | **Train** | **55.43** | 54.29 | 0 | 47.06 | 40.64 | 73.2 | 81.96 | 92.85 | 84.37 | 92.91 | 87.75 | On sensor 3 only | |
| **Test** | **55.42** | 54.07 | 0 | 47.32 | 39.73 | 73.34 | 82.03 | 92.85 | 84.51 | 92.9 | 87.57 |
| **Model 9** | **Train** | **55.43** | 54.29 | 50.67 | 47.06 | 40.64 | 73.2 | 81.96 | 92.85 | 84.37 | 92.91 | 87.75 | On sensor 4 only | |
| **Test** | **62.49** | 71.20 | 53.87 | 50.42 | 58.37 | 68.54 | 91.89 | 94.75 | 82.85 | 93.46 | 87.35 |
| **Model 10** | **Train** | **75.21** | 95.62 | 91.33 | 63.33 | 55.54 | 73.79 | 99.70 | 93.22 | 94.99 | 94.97 | 87.06 | On sensor 1,2,3 | |
| **Test** | **74.61** | 5.71 | 33.68 | 62.46 | 53.7 | 73.56 | 99.73 | 93.17 | 95 | 94.85 | 86.54 |

Next we will build stepAIC model to see the important variables.

**3.2 stepAIC Model:**

stepAIC selects the model based on Akaike Information Criteria, not p-values. The goal is to find the model with the smallest AIC by removing or adding variables in your scope.

Performs stepwise model selection by AIC.

**Table: stepAIC Models Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | **Specificity (%)** | | | | | **Remarks** |
| **Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |
| **stepAIC**  **Model 1** | **Train** | **83.27** | 97.81 | 73.67 | 76.08 | 72.98 | 79.03 | 99.78 | 96.99 | 93.06 | 96.75 | 92 | All data (user and sensor data) |
| **Test** | **83.3** | 97.87 | 74.89 | 75.84 | 73.86 | 78.89 | 99.83 | 97.01 | 92.93 | 96.78 | 92.06 |
| **stepAIC**  **Model 2** | **Train** | **82.68** | 97.70 | 71.13 | 72.88 | 72.84 | 82.91 | 99.75 | 96.85 | 95.02 | 96.72 | 90 | Only on sensor data- all 4 accelerometer data |
| **Test** | **82.62** | 97.80 | 72.57 | 72.38 | 73.52 | 82.51 | 99.81 | 96.82 | 94.48 | 96.76 | 89.87 |

By building stepAIC model on 2 data-set one is on whole data and other on sensor data we found that it give same result as it gives on multi-nomial.

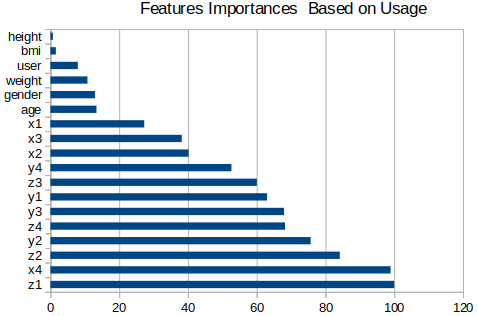
Here, we found it does not help much in finding the important feature as it take all features as important.

**3.3 Decision Tress (C5.0):**

C5.0 algorithm is widely used as a decision tree method in machine learning. Initially we have ID3.0 algorithm. Based on ID3.0, people developed C4.5 algorithm, and finally develop C5.0 algorithm. C5.0 algorithm to build either a decision tree or a rule set. This type of decision tree model is based on entropy and information gain.

A C5.0 model works by splitting the sample based on the field that provides the maximum information gain. Each subsample defined by the first split is then split again, usually based on a different field, and the process repeats until the sub samples cannot be split any further. Finally, the lowest-level splits are reexamined, and those that do not contribute significantly to the value of the model are removed or pruned.

The C5.0 node can predict only a categorical target. And we have categorical target to classify thus we go with this and see how it is behaving. In R, the C50 package is used to get C5.0 algorithm in classification.



By default, C5.0 measures predictor importance by

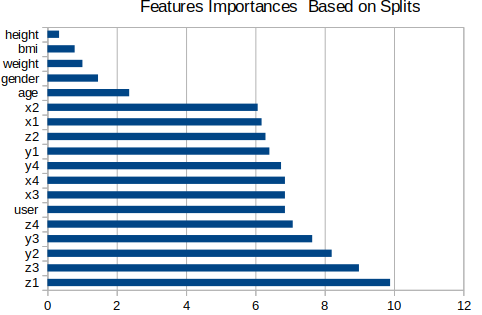
determining the percentage of training set samples

that fall into all the terminal nodes after the split

(this is used when metric = "usage"). For

example, predictor in the firs t split automatically

has an importance measurement of 100 percent.

Other predictors may be used frequently in splits,

but if the terminal nodes cover only a handful of

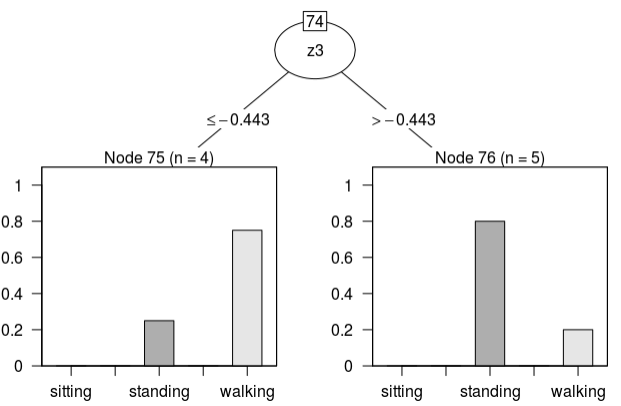
training set samples, the importance scores may

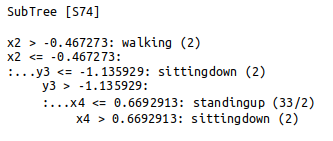
be close to zero. When metric = "splits",

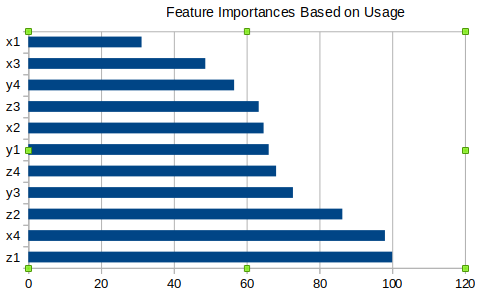
the percentage of splits associated with each

predictor is calculated.

In contrast, a rule set is a set of rules that tries to make predictions for individual records. Rule sets are derived from decision trees and, in a way, represent a simplified or distilled version of the information found in the decision tree. Rule sets can often retain most of the important information from a full decision tree but with a less complex model. Because of the way rule sets work, they do not have the same properties as decision trees. The most important difference is that with a rule set, more than one rule may apply for any particular record, or no rules at all may apply. If multiple rules apply, each rule gets a weighted "vote" based on the confidence associated with that rule, and the final prediction is decided by combining the weighted votes of all of the rules that apply to the record in question. If no rule applies, a default prediction is assigned to the record.

Partition sub-trees (partial result) :





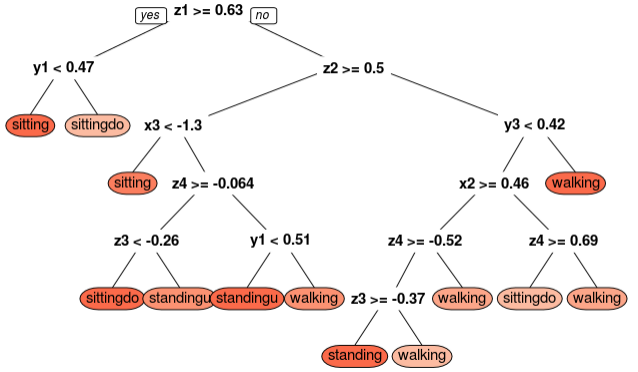
Feature importance on sensor data only. Waist acceerometer is most important feature in recognizing standing, walking and sitting classes.

Arm accelerometer is most important feature to

recognize sitting-down class. Then thigh and ankle

accelerometer sensor with the other 2 acceleromter sensor data combinedly recognize all classes accurately.

The decision tree visualization show here,indicates its structure. It shows the attribute’s selection order for criterion as information gain



**Table: Decision Tree C50 Models Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | **Specificity (%)** | | | | | **Remarks** |
| **Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |
| **Decision Tree C50 Model1** | **Train** | **99.42** | 99.97 | 98.01 | 99.63 | 97.73 | 99.39 | 99.95 | 99.87 | 99.83 | 99.84 | 99.78 | Only on sensor data  (waist, ankle, arm and z2 of thigh) |
| **Test** | **97.97** | 99.89 | 94.1 | 98.78 | 92.47 | 97.47 | 99.88 | 99.53 | 99.3 | 99.47 | 99.29 |
| **Decision Tree C50 Model2** | **Train** | **96.9** | 99.72 | 89.06 | 98.68 | 88.67 | 96.15 | 99.81 | 99.24 | 99.01 | 99.33 | 98.72 | Only on top 4 important features  z1,x4,z2,y3 |
| **Test** | **94.41** | 99.37 | 81.11 | 97.3 | 79.72 | 93.29 | 99.66 | 98.57 | 98.38 | 98.64 | 99.79 |
| **Decision Tree C50 Model3** | **Train** | **98.2** | 99.91 | 93.98 | 99.01 | 93.26 | 97.89 | 99.94 | 99.56 | 99.45 | 99.59 | 99.21 | Only on 2 accelerometer : Waist & Ankle |
| **Test** | **95.82** | 99.82 | 85.98 | 97.69 | 84.36 | 95.5 | 99.89 | 98.95 | 98.87 | 98.94 | 98.18 |

To recognize all the classes accurately we use 4 acclerometers. And model 1 is performing better with an accuracy of 99.41% on train data and 97.97% on test data.

Next we will try svm, which try to separate the classes by projecting in higher dimension and see with which kernel, cost and gamma combination it accurately classify the classes.

**3.4 SVM:**

SVM used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).

The SVM algorithm is implemented in practice using a kernel.

**3.4.1 SVM on whole data: Radial Kernel**

Finally, we can also have a more complex radial kernel. For e.g : K(x,xi) = exp(-gamma \* sum((x – xi^2))

Where gamma is a parameter that must be specified to the learning algorithm. A good default value for gamma is 0.1, where gamma is often 0 < gamma < 1. The radial kernel is very local and can create complex regions within the feature space, like closed polygons in two-dimensional space.

**Summary of svm model:**

svm(formula = class ~ ., data = train\_data, kernel = "radial")

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

gamma: 0.1

Number of Support Vectors: 14029

**3.4.2 SVM on whole data: Linear Kernel**

The dot-product is called the kernel and can be re-written as: K(x, xi) = sum(x \* xi)

The kernel defines the similarity or a distance measure between new data and the support vectors. The dot product is the similarity measure used for linear SVM or a linear kernel because the distance is a linear combination of the inputs.

**3.4.3 SVM on whole data: Polynomial Kernel**

Instead of the dot-product, we can use a polynomial kernel, for example:

K(x,xi) = 1 + sum(x \* xi)^d

Where the degree of the polynomial must be specified by hand to the learning algorithm. When d=1 this is the same as the linear kernel. The polynomial kernel allows for curved lines in the input space.

**Table: SVM Models Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | | **Specificity (%)** | | | | | | **Remarks** |
| **SVM Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |  | |
| **Radial**  **Model 1** | **Train** | **97.81** | 99.78 | 93 | 96.37 | 95.9 | 99.01 | 99.95 | | 99.65 | 99.74 | 99.3 | 98.66 | Only on sensor data -all 4 accelerometer reading except x2,y2  by using Radial kernel | |
| **Test** | **97.86** | 99.72 | 93.07 | 96.46 | 95.86 | 99.21 | 99.94 | | 99.7 | 99.78 | 99.27 | 99.21 |
| **Linear**  **Model 2** | **Train** | **821.89** | 97.35 | 59.66 | 72.49 | 68.67 | 87.33 | 99.69 | | 97.42 | 96.26 | 95.91 | 88.51 | Only on sensor data -all 4 accelerometer reading except x2,y2  by using Linear kernel | |
| **Test** | **82.17** | 97.57 | 59.83 | 72.97 | 72.97 | 68.67 | 99.72 | | 97.45 | 96.24 | 95.84 | 88.85 |
| **Polynomial**  **Model 3** | **Train** | **93.87** | 99.60 | 90.55 | 85.47 | 95.64 | 98.93 | 99.74 | | 99.03 | 99.82 | 98.27 | 95.69 | Only on sensor data -all 4 accelerometer reading except x2,y2  by using Polynomial kernel | |
| **Test** | **93.72** | 99.57 | 90.51 | 85.34 | 94.61 | 94.61 | 98.84 | | 99.67 | 99.05 | 99.79 | 95.66 |

We have build svm model using 3 kernels one by one and found svm with radial kernel perform better among all. Even in higher dimension it does not able to linearly separate the data. By using polynomial also it perform a quite well as compared to linear model.

**3.5 Naive Bayes Model:**

Naive Bayes Model for discrete predictors. Bayesian classification can predict class membership probabilities. The effect of an attribute value on a given class is independent of the value of the other attributes is assumed by the Naive Bayes algorithm . The Naive Bayes algorithm scales continuously in the number of predictors and rows and builds rapidly models. Naive Bayes algorithm derives the probability of a prediction.

The probability of event X occurring given that event Y has occurred (P(X|Y)) is proportional to the probability of event Y occurring given that event X has occurred multiplied by the probability of event X occurring ((P(Y|X)P(X)).

**Table: Naive Bayes Model Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | **Specificity (%)** | | | | | | **Remarks** |
| **Naive Bayes Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |  | |
| **Model 1** | **Train** | **77.23** | 91.30 | 58.37 | 92.54 | 13.5 | 67.49 | 97.19 | 96.27 | 82.51 | 97.86 | 96.07 | Only on sensor data -all 4 accelerometer reading except x2,y2 | |
| **Test** | **77.20** | 91.30 | 58.35 | 92.37 | 13.21 | 67.65 | 97.09 | 96.43 | 82.25 | 97.98 | 96.03 |

Naive Bayes Model for classifying the human activities does not performed well.

**3.6 Knn Model:**

KNN has no model other than storing the entire data set, so there is no learning required. Efficient implementations can store the data using complex data structures like k-d trees to make look-up and matching of new patterns during prediction efficient.Because the entire training data set is stored, you may want to think carefully about the consistency of your training data. It might be a good idea to curate it, update it often as new data becomes available and remove erroneous and outlier data.

The choice of K is essential in building the KNN model. One appropriate way to look at the number of nearest neighbors k is to think of it as a smoothing parameter. For any given problem, a small value of k will lead to a large variance in predictions. Alternatively, setting k to a large value may lead to a large model bias. Thus, k should be set to a value large enough to minimize the probability of misclassification and small enough (with respect to the number of cases in the example sample) so that the K nearest points are close enough to the query point.

Knn model’s accuracy on train data is 94.72% and on test it is 94.74% with k= 340. So here the classification are based on majority of voting of the outcomes of the K-nearest neighbors.

**Table: Naive Bayes Model Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | | **Specificity (%)** | | | | | | **Remarks** |
| **Knn Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |  | |
| **Knn Model 1** | **Train** | **94.72** | 99.66 | 92.08 | 99.36 | 76.45 | 89.83 | 99.45 | | 98.41 | 96.15 | 99.59 | 99.58 | Only on sensor data -all 4 accelerometer reading except x2,y2 by using k =340 nearest neighbor | |
| **Test** | **94.74** | 99.66 | 91.2 | 99.43 | 77.05 | 89.9 | 99.41 | | 98.58 | 96.11 | 99.55 | 99.54 |

**3.7 Random Forest Model:**

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Say, we have 1000 observation in the complete population with 10 variables. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process (say) 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction.

Random forests does not over fit. You can run as many trees as you want. It is fast. Running on a data set with 50,000 cases and 100 variables, it produced 100 trees in 11 minutes on a 800Mhz machine. For large data sets the major memory requirement is the storage of the data itself, and three integer arrays with the same dimensions as the data. If proximities are calculated, storage requirements grow as the number of cases times the number of trees.

When the training set for the current tree is drawn by sampling with replacement, about one-third of the cases are left out of the sample. This oob (out-of-bag) data is used to get a running unbiased estimate of the classification error as trees are added to the forest. It is also used to get estimates of variable importance. The out-of-bag (oob) error estimate.

In random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows: Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation** | **Data Set** | **Accuracy**  **(%)** | **Sensitivity (%)** | | | | | | **Specificity (%)** | | | | | | **Remarks** |
| **Random Forest Model** | **Sitting** | **Sitting-down** | **Standing** | **Standing-up** | **Walking** | **Sitting** | | **Sitting-down** | **Standing** | **Standing-up** | **Walking** |  | |
| **RF Model 1** | **Train** | **99.99** | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 100 | 100 | 100 | Only on sensor data -all 4 accelerometer reading except x2,y2  r | |
| **Test** | **99.49** | 99.94 | 98.54 | 99.7 | 97.7 | 97.22 | 99.65 | | 99.99 | 96.87 | 99.88 | 99.78 |

**Conclusion and Future Work**

C5 is a classifier which classifies the data in less time compare to other classifier. For generating decision tree the memory usage is minimum and it also improve the accuracy. This proposed system is developed on the bases of C5 algorithm. In the proposed system C5.0 algorithm provides Feature selection, Cross validation and reduced error pruning facilities. So the further scope of this algorithm is achieved by implementation of new features like PCA, Cross Validation and Model Complexity.

Random Forest can be used and it perform well in recognizing the human activities with an accuracy with 99.49%.

Furthermore, this work shows acceleration can be used to recognize a variety of household activities for context-aware computing. This extends previous work on recognizing ambulation and posture using acceleration (see Figure 1).

This work further suggests that a mobile computer and small wireless accelerometers placed on an individual’s thigh and dominant wrist may be able to detect some common everyday activities in naturalistic settings using fastFFT-based feature computation and a decision tree classifier algorithm. Decision trees are slow to train but quick to run. Therefore, a pre-trained decision tree should be able to classify user activities in real-time on emerging mobile computing devices with fast processors and wireless accelerometers.

In this experiment to recognize these classes these 4 accelerometers are needed but in future to recognize new classes and investigate the classifier’s performance with the use of accelerometers in different positions and in different quantities will be needed.

**Future Work :** Smart phones are ubiquitous and becoming more and more sophisticated.This has been changing the landscape of people’s daily life and has opened the doors for many interesting data mining applications. Human activity recognition is a core building block behind these applications. It takes the raw sensor’s reading as inputs and predicts a user’s motion activity. This paper presents a comprehensive survey of the recent advances in activity recognition with smart phone's sensors. Here introduce the basic

concepts of activity recognition (such as sensors, activity types, etc.).

Here review the core data mining techniques behind the mainstream activity recognition algorithms, analyze their major challenges, and introduce a variety of real applications enabled by activity recognition. The activity recognition based on smart phone sensors leads to many possible future research directions. Besides the applications mentioned in Section V, an even novel way could be equipping smart phones with intelligent applications to replace the traditional devices such as remote control,traffic controlling, and tracking devices. Smart phone applications that can recognize users’ gestures could send a corresponding command to home electronics. Thus, instead of keeping different remotes in one’s cabinet, can just install one application that has the remote functions. The cross field research could be developed in many fields because of the mobile activity recognition techniques.

**Reference:**

[http://www.pervasive.jku.at/Teaching/\_2012SS/EmbeddedSystems/Uebungen/UE52/2004\_Activity%20Recognition%20from%20User-Annotated%20Acceleration%20Data\_Intille.pdf](http://www.pervasive.jku.at/Teaching/_2012SS/EmbeddedSystems/Uebungen/UE52/2004_Activity Recognition from User-Annotated Acceleration Data_Intille.pdf)

International Journal for Innovative Research in Science & Technology September 2015

<http://www.ijirst.org/articles/IJIRSTV2I4024.pdf>

<http://archive.ics.uci.edu/ml/datasets/Wearable+Computing%3A+Classification+of+Body+Postures+and+Movements+%28PUC-Rio%29>

<http://groupware.les.inf.puc-rio.br/work.jsf?p1=10335>

<http://groupware.les.inf.puc-rio.br/public/papers/2012.Ugulino.WearableComputing.HAR.Classifier.RIBBON.pdf>