HUMAN ACTIVITY RECOGNITION USING FEATURE SELECTION

Scott Onestak

Introduction

A Samsung smart phone with an embedded accelerometer and gyroscope is able to capture 561 different features from the subject possessing it [A]. These features can then be used to predict everyday activities such as walking, walking upstairs, walking downstairs, sitting, standing, and laying, but the copious number of features makes real time prediction of these everyday activities too costly to calculate if all features are used. Therefore, the task is to find a predictive model that uses the fewest amount of features while still attaining an accuracy of 80 and 90 percent.

It is hypothesized that by using a subset of features, selected by the random forest algorithm as the most important features, a model can be constructed through which the desired predictive accuracy can be reached. This was proved true. Using a random forest model with only the selected features as dependent variables, an accuracy of 80 percent could be reached with three features and 90 percent with five.

Data Collection and Preparation

The data, including the 561 features, subject number, and activity for 7,352 observations, was provided through an .rda file via Blackboard. No missing values were found in the dataset, and all feature values were already normalized to values between -1 and 1. Once this .rda file was loaded into RStudio, the activity variable was immediately changed into a factor variable because the observations are categorical. After inspection of the dataset, it was discovered that some features shared the same name. Therefore, those sharing the same feature name were renamed so that no features were named the same [B].

In preparing the data for training and testing, the caret package was used to partition the data into the training and test sets, which consisted of 80 percent and 20 percent of the data respectively. The partitioning function creates a stratified split in which the same proportion of the activity factors are randomly selected for the training and testing sets.[1] This is done so that the training set does not disproportionately train on certain activities over the others, which could cause bias when features are selected because they would not be representative of the dataset.

After, the subject number is dropped from the training and testing sets so that it may not be used as a variable in the model. This is because there is no logical justification as to why the subject’s unique identification number should correlate to the observed activity. While it may be true that variations in individuals’ activities occur, there would be no way to correlate future

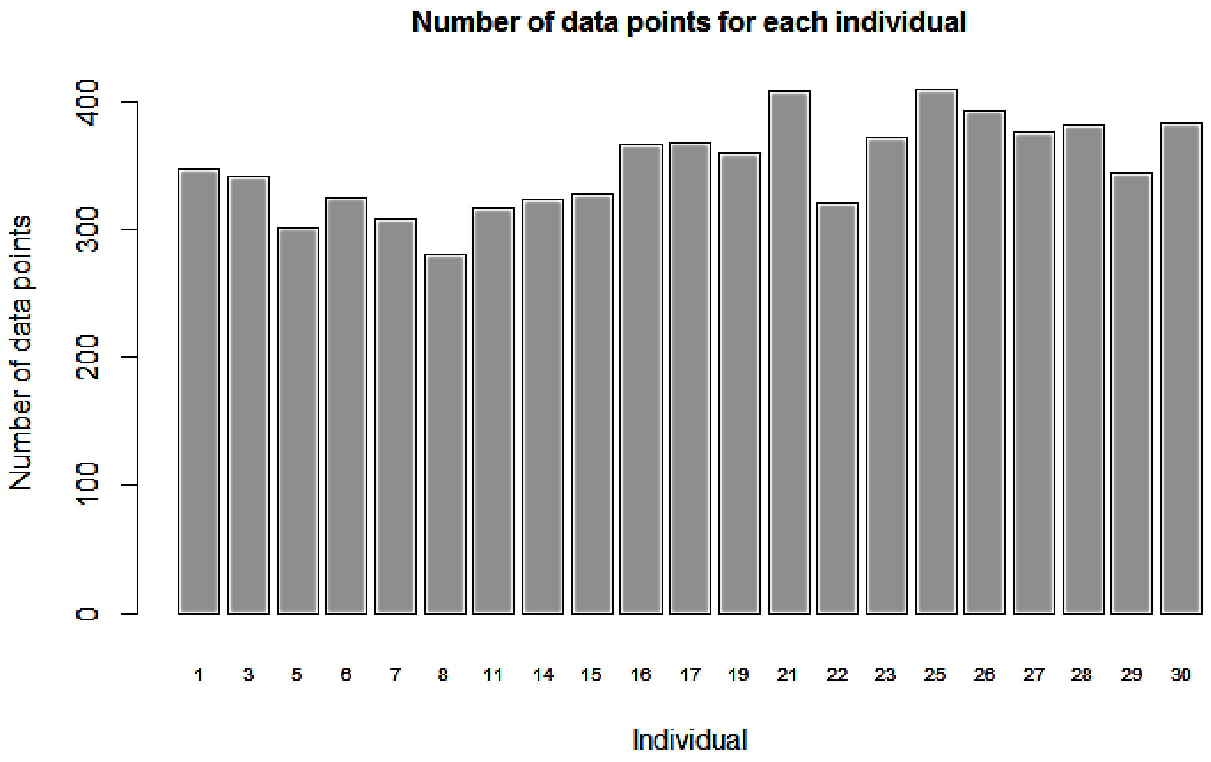
[1] Kuhn, Max. "A Short Introduction to the Caret Package." August 6, 2015. Accessed October 3, 2016. https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf.

users' subject identification numbers to their activities, unless through some spurious correlation. In which case, the feature would not be correlated through scientific observation, but sheer randomness.

Exploratory Analysis

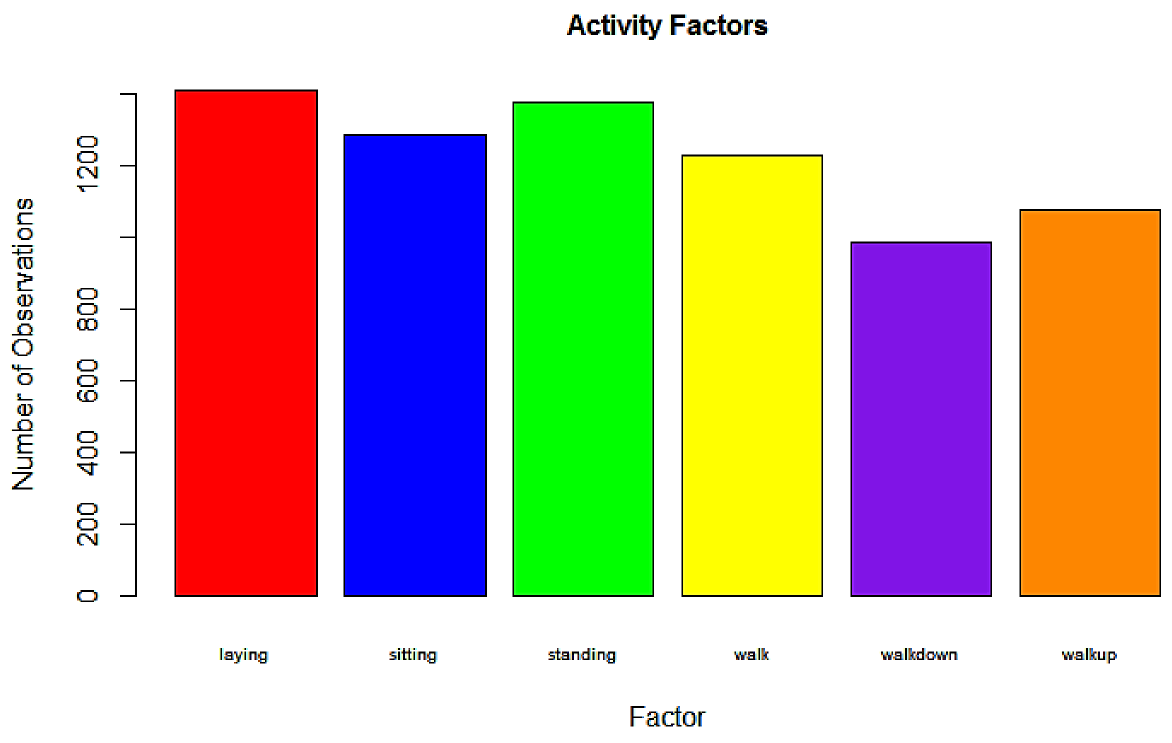
There were many ways in which the data was explored; the first being through the simple number of data points each individual subject had. Though not completely consistent across all individuals, the data shows a relatively even number of data points from each person, which can be observed in Figure 1 below.

Figure 1



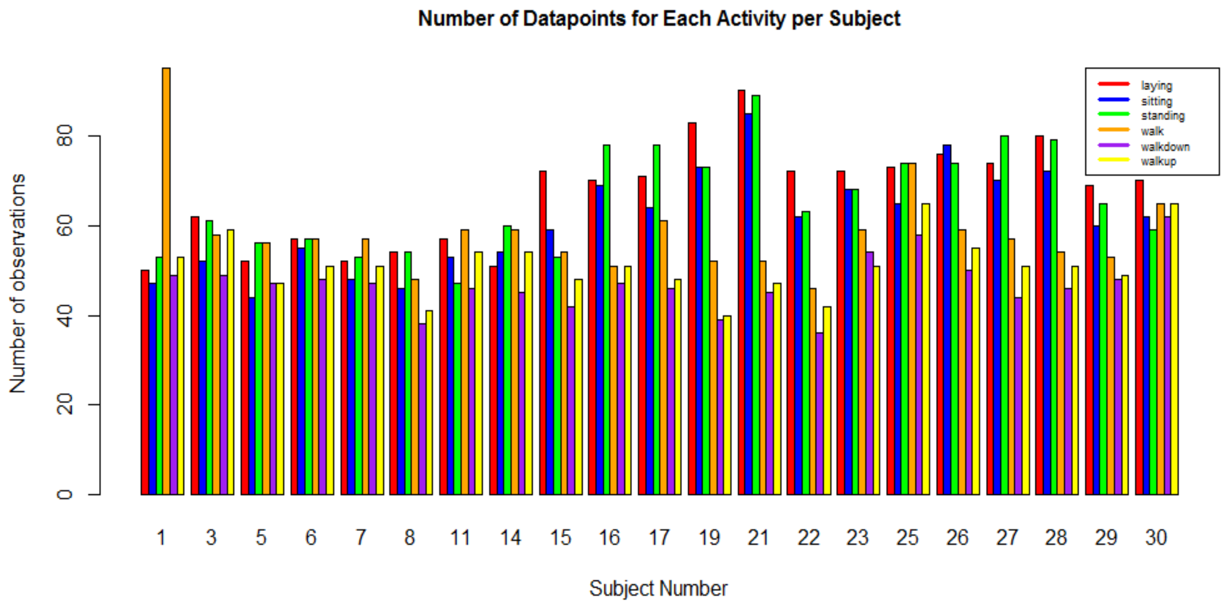
Then, the number of activity attributes themselves were observed. Though each factor was not observed consistently, the decent amount of observations for each of the factors makes it less likely random noise in the data will affect the features selected when it comes time to choose the most important splitting features. The number of observations for each activity can be observed in Figure 2.

Figure 2



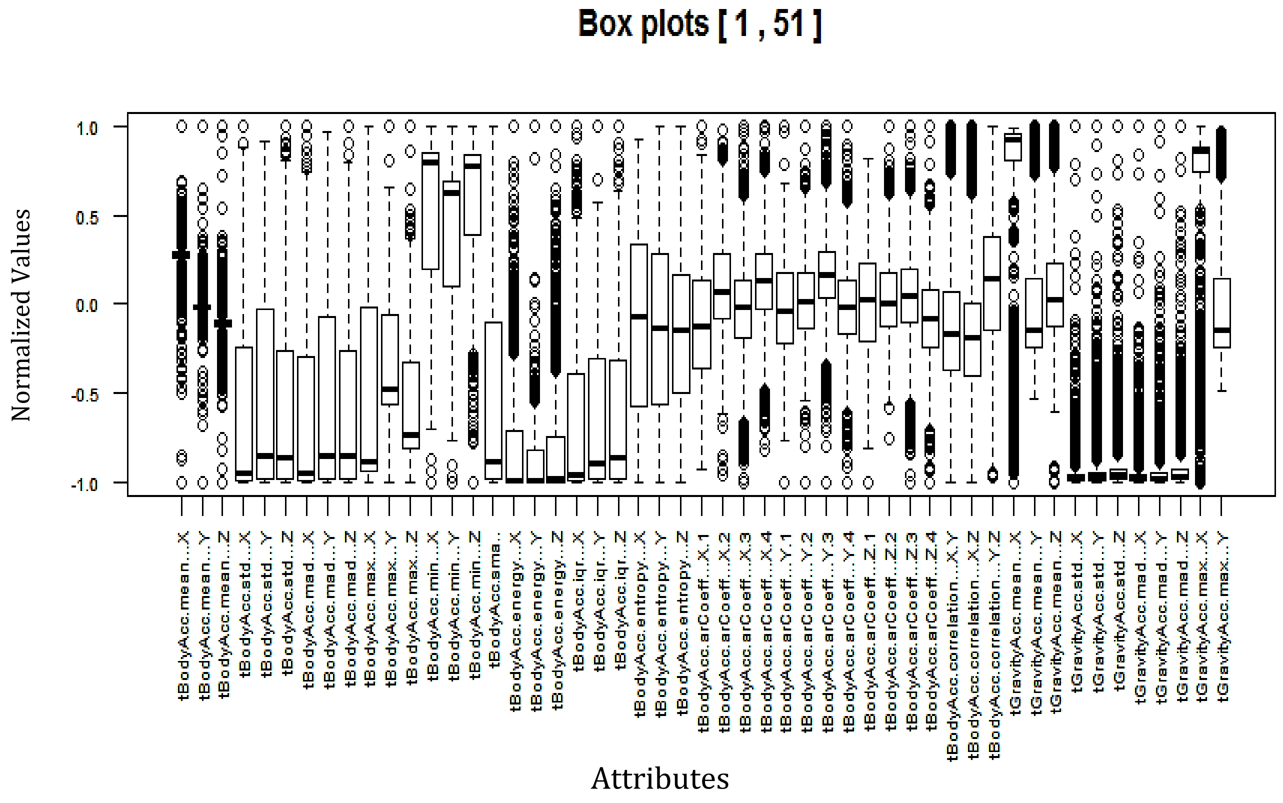
Next, each activity for each individual was graphed in order to determine that every person had a relatively equal amount of observations for each activity. If the data had only received "walk" information from one individual and "laying" from another, instead of receiving every attribute from all people, then this may have biased the data due to individual differences between people. However, each individual had a decent number of observations for each activity, which can be observed below in Figure 3.

Figure 3



Finally, we observed each of the individual features that may be selected in the feature selection. To do this, a box-and-whisker plot was created for each feature. These were graphed 50 at a time; an example is provided in Figure 4. The entirety of these box-and-whisker plots can be observed in Appendix C.

Figure 4



From observing the box-and-whisker plots, the skewness of the features and how that would affect feature selection needed to be analyzed. It was observed that the skew of the features fell into three different categories: (1) normally distributed, (2) skewed, and (3) highly skewed. Therefore, graphs for each of these types were analyzed in order to see which may produce better splitting features. These graphs can be observed in Appendix D. We hypothesized that skewed and highly skewed features posed as better candidates for feature selection because they can segment off certain activities based on different values; whereas, normally distributed features tend to cluster around the same value no matter the activity. Overall, there were 284 normally distributed, 111 skewed, and 166 highly skewed features in the dataset.

Methods

In order to compute the most important features in the dataset, a random forest algorithm with 5-fold cross-validation using parallel processing was employed on the training set using the caret package [E]. The assumption that underlies the random forest model is that the data sampled for the training set is representative of the overall data.[2] This assumption is therefore met during the data preparation stage, where the training and test sets were formed proportionally to the activity factors.

From this algorithm, the features were selected by using the varImp function as a wrapper around the random forest algorithm in order to rank the features by their importance.[3] While it is almost impossible to determine the exact combination of features that when coupled in the model will be the optimal solution to the problem, it can be assumed that the most important features individually would provide superior classification when paired together. Therefore, features are selected from the most important to least important until the threshold measures of 80 and 90 percent are met.

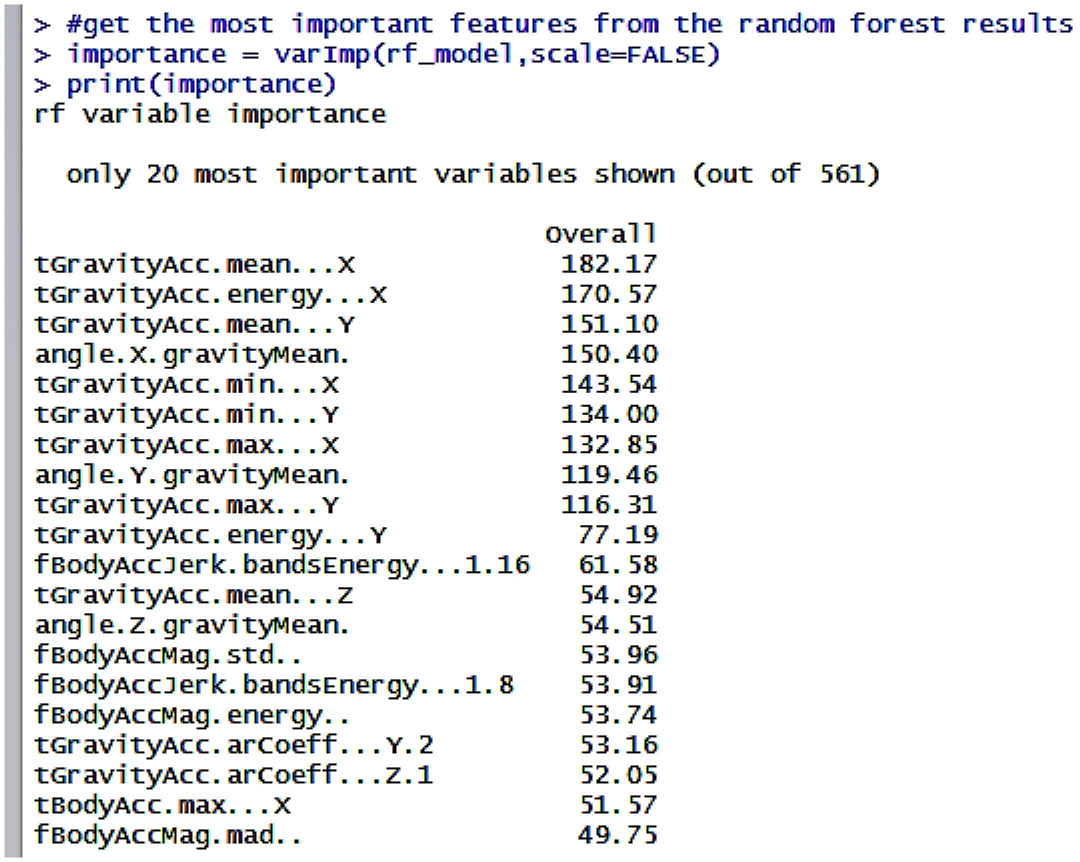
These selected features are then applied to a random forest algorithm on the training set iteratively adding the next best feature until the thresholds were reached in order to estimate how many features would be necessary in order to predict the number of features needed to reach the accuracy thresholds on the testing set [F]. After the number of features were observed, these features were applied iteratively to the test set in the same way in order to predict the activity via the same random forest model.

[2] "Assumptions/Limitations of Random Forest Models." Stack Exchange. Accessed October 04, 2016. http://datascience.stackexchange.com/questions/6015/assumptions-limitations-of-random-forest-models.
[3] Kuhn, Max. "A Short Introduction to the Caret Package." August 6, 2015. Accessed October 3, 2016. https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf.

Analysis and Results

Applying 5-fold cross-validation to the training set via the random forest algorithm of caret package, the top features from the model were obtained, as shown in Figure 5.

Figure 5



```
> #get the most important features from the random forest results
> importance = varImp(rf_model,scale=FALSE)
> print(importance)
rf variable importance

  only 20 most important variables shown (out of 561)

                                Overall
tGravityAcc.mean...X             182.17
tGravityAcc.energy...X           170.57
tGravityAcc.mean...Y             151.10
angle.X.gravityMean.             150.40
tGravityAcc.min...X              143.54
tGravityAcc.min...Y              134.00
tGravityAcc.max...X              132.85
angle.Y.gravityMean.             119.46
tGravityAcc.max...Y              116.31
tGravityAcc.energy...Y            77.19
fBodyAccJerk.bandsEnergy...1.16   61.58
tGravityAcc.mean...Z              54.92
angle.Z.gravityMean.              54.51
fBodyAccMag.std..                 53.96
fBodyAccJerk.bandsEnergy...1.8    53.91
fBodyAccMag.energy..              53.74
tGravityAcc.arCoeff...Y.2         53.16
tGravityAcc.arCoeff...Z.1         52.05
tBodyAcc.max...X                  51.57
fBodyAccMag.mad..                 49.75
```

These features were then applied iteratively to the random forest algorithm in order to obtain the accuracy of the model. Under this process, the first model predicts activity using the function

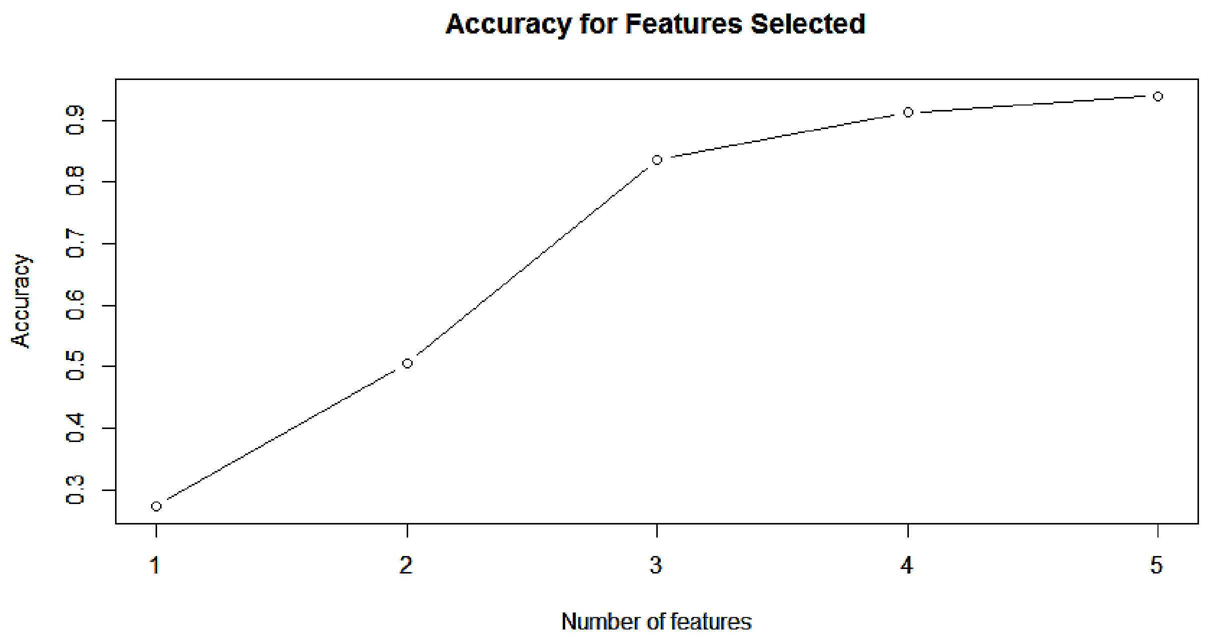
$$activity \sim tGravityAcc.mean \dots X,$$

the second using the function

$$activity \sim tGravityAcc.mean \dots X + tGravityAcc.energy \dots X,$$

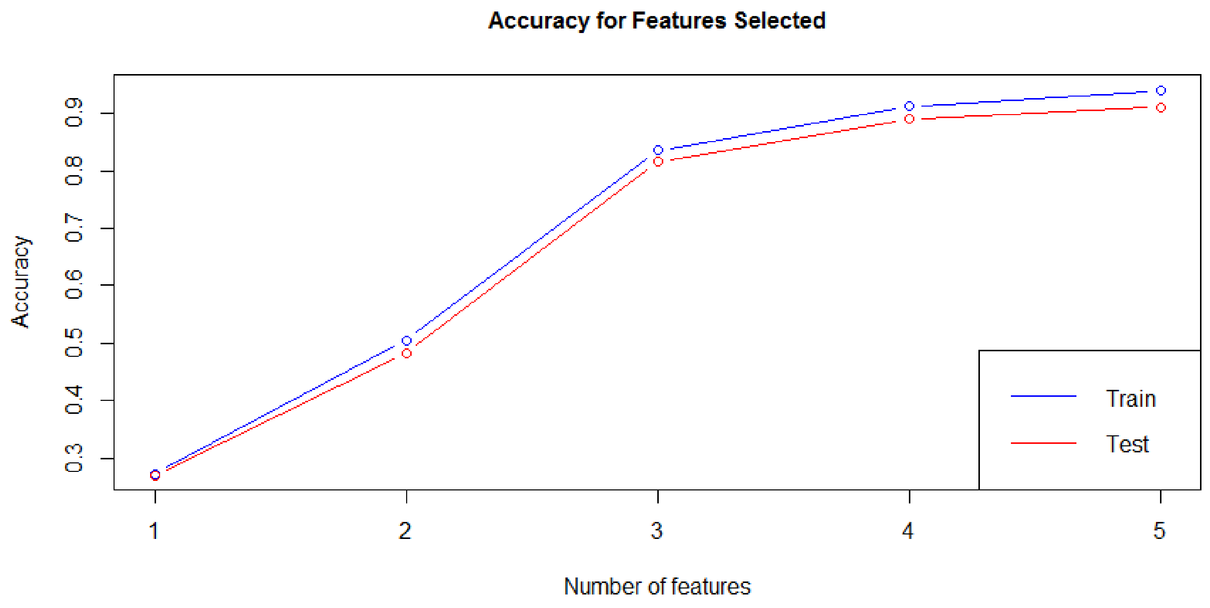
and so on until the features reached the threshold of 90 percent. The model reached the accuracy of 80 percent after three features and 90 percent after the top four features. Because the four-feature model was slightly above the 90 percent threshold, a five-feature model was also calculated on the training set if necessary for the test set. These accuracies for the prediction of the activity feature were then graphed in order to observe the information gained from the addition of each feature, as seen in Figure 6.

Figure 6



These random forest models, when applied to the test set, produced similar results to the training set. Eighty percent accuracy was achieved through the top three features, tGravityAcc.mean…X, tGravityAcc.energy…X, and tGravityAcc.mean…Y, and the model reached 90 percent accuracy with the top five features, tGravityAcc.mean…X, tGravityAcc.energy…X, tGravityAcc.mean…Y, angle.X.gravityMean., and tGravityAcc.min…X. These were then graphed in comparison to the training set as shown below in Figure 7.

Figure 7

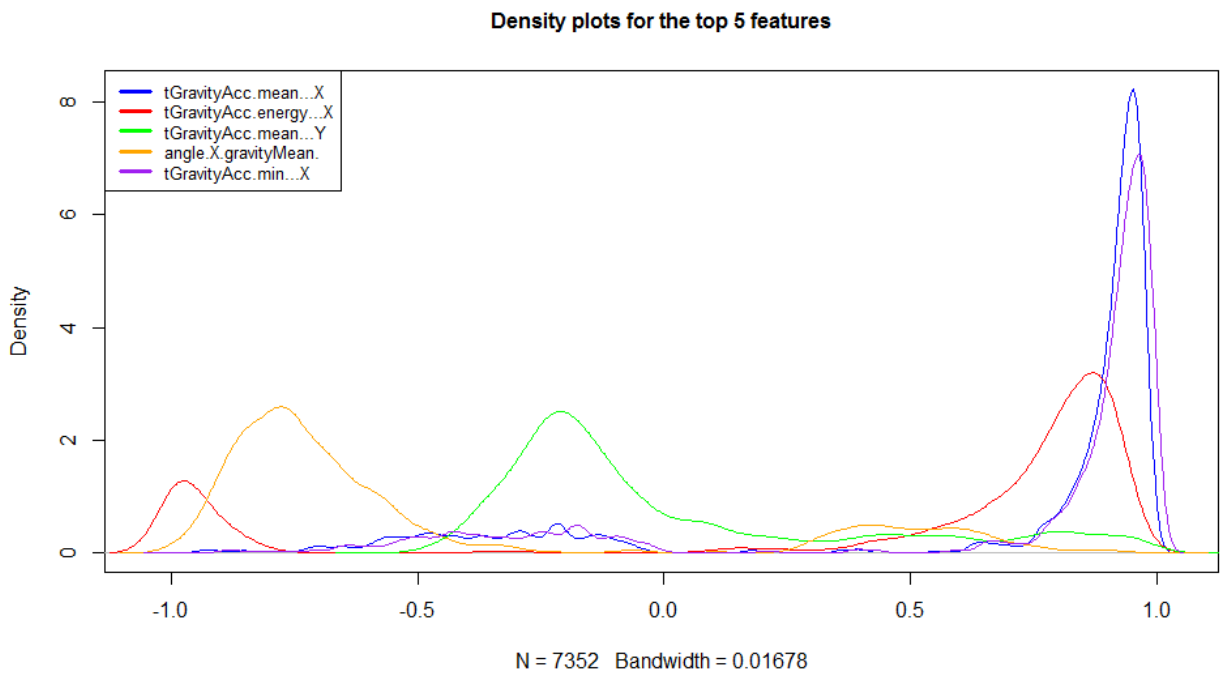


With the five-feature model, the training set achieved 94.02 percent accuracy while the test set was accurate 91.14 percent of the time, suggesting a slight deviation between the accuracy on the training and test sets. This was consistent across all the models [G], but for the most part, the random forest model accurately fit the data across the feature selection subsets.

Analyzing the features selected, four of the top five attributes measure the gravity acceleration along one of the axes, which probably helped the algorithm separate the activities based on acceleration speed. The other was an angle feature, which probably helped the algorithm separate activities such as "walk," "walkup," and "walkdown" because the person's body is moving at a different angle.

However, the most telling statistic of the top selected features is that all of them are skewed features of the three earlier categories: normally distributed, skewed, and highly skewed [H]. Observing the graphs of these features, which can be seen in Figure 8, it appears that these skewed features might be better at distinguishing between the activities because most of them possess multiple peaks within their density distributions. Therefore, different activity factors may have been easier to separate based on where their information fell within these distributions with multiple groupings of observations.

Figure 8



Lastly, the results of the feature selection subsets can be compared to the accuracy achieved using the complete feature set. Using all 561 variables on the training set, the accuracy of the random forest model is 98.10 percent. Similarly, the accuracy of these features on the test set is 96.12 percent. Therefore, the accuracy of the five-feature subset is about five percent less accurate than the use of all features. This shows that beyond those five features, not much more information can be gained from additional features, and by limiting the features selected, it allows the random forest algorithm to be able to quickly and accurately calculate the activity in real time.

Conclusions

In conclusion, a random forest algorithm was able to predict the activity with 80 percent accuracy using three features and 90 percent accuracy using five. While these may not be the best combination of features used from the dataset, the features prove superior to others based on their individual importance in predicting the activity factor.

The random forest feature selection via the caret package proved successful in generating these features as opposed to an earlier used algorithm from the Biocomb package that proved less successful. The drawback to using this random forest algorithm was the runtime, which lasted approximately 20 minutes. A potential solution to this in the future may be to reduce the number of features able to be selected by the random forest algorithm to those that are of the skewed type since they appear to be the most successful features in predicting the activity. However, further analysis would need to be conducted before jumping to such a conclusion.

Some cause for concern may be the slight difference in accuracy between the training and test sets, which may imply a slim overfitting of the data. This could potentially be due to the fact that when randomly assigning observations to the training and test sets, the observations were only split taking into consideration the activity feature; it may also be important to factor in the subject when splitting so that the individual differences of each person are split evenly between the training and test data.

Overall, the random forest model proved to be an accurate and, with a subset of features, speedy enough model to predict the activity in real time. By observing the importance of the individual features, a subset of highly predictive features is able to accurately predict the activity factor 80 percent of the time with three feature and 90 percent with five.

Appendix A

Samsung smart phone data collection

Thirty participants between the ages of 19 and 48 wore a Samsung Galaxy S II on their waist as they performed six different daily activities: walking, walking upstairs, walking downstairs, sitting, standing, and laying. The embedded accelerometer and gyroscope capture 3-axial angular velocity at a constant rate of 50 Hz.[4] The accelerometer and gyroscope signals were "pre-processed by applying noise filters and then sampling in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window)."[5] Body motion and gravitational components were then separated using a Butterworth low-pass filter where a 0.3 Hz cutoff frequency was used. A vector of features, which were used in this problem, were obtained by calculations from the time and frequency domains.[6]

Appendix B

The Renaming of features sharing the same name

Features sharing the same name were renamed by the following procedure. First, a table of names was constructed to observe the frequency of each name. Then, R's make.names function was employed to create unique names for the features. For example, the three features named "fBodyAccJerk.bandsEnergy...9.16" became "fBodyAccJerk.bandsEnergy...9.16," "fBodyAccJerk.bandsEnergy...9.16.1," and "fBodyAccJerk.bandsEnergy...9.16.2." These changes were then applied to the original dataset. It should be noted that none of these renamed features was selected by the feature selection algorithm, so the specific names given to these features is not necessary to know for the problem at hand.

[4] Reyes-Ortiz, Jorge L., Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. *README*. Txt. November 2013.
[5] Ibid.
[6] Ibid.

Appendix C

Box-and-whisker plots for each feature of the dataset

Figure 9

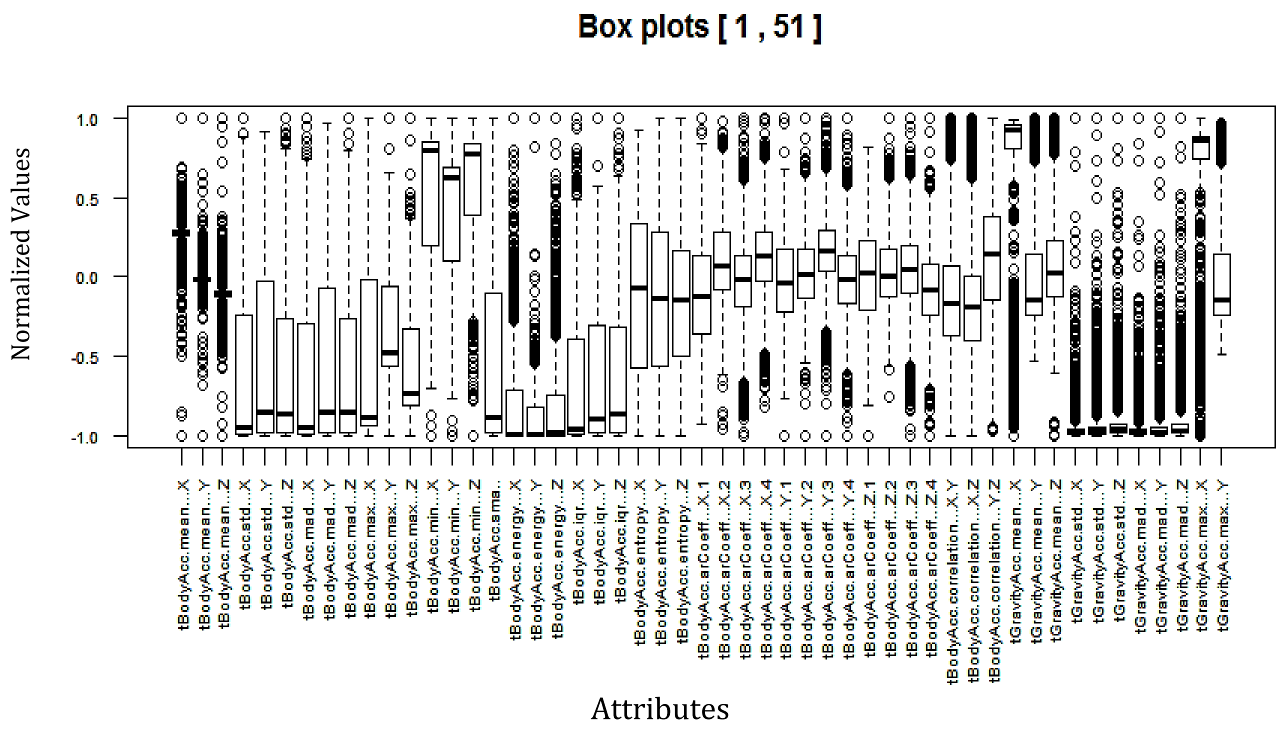


Figure 10

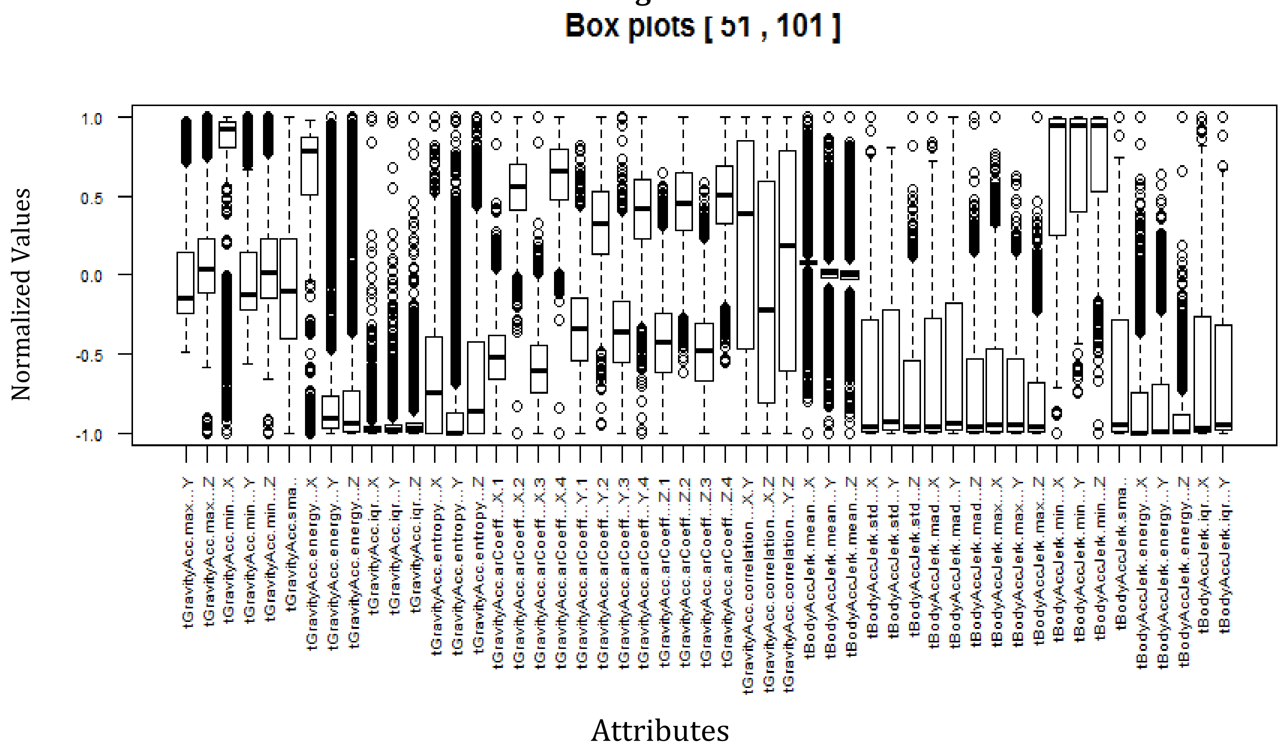


Figure 11

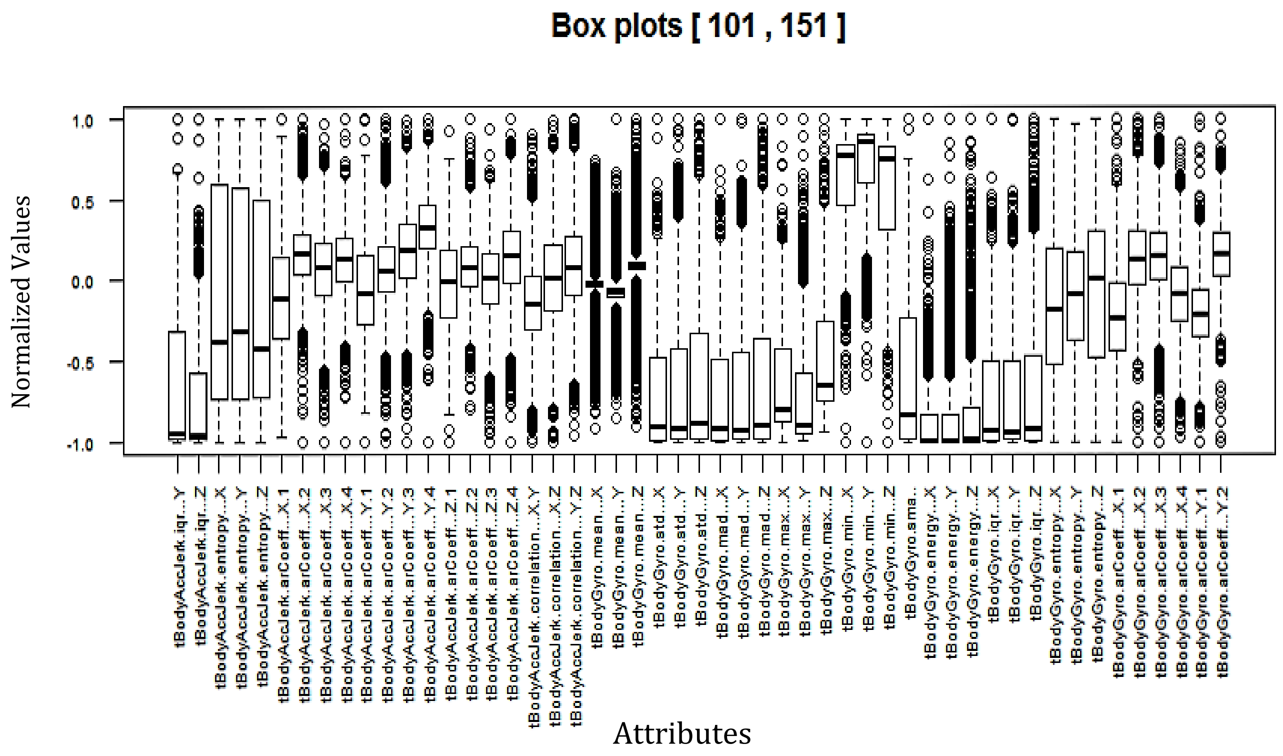


Figure 12

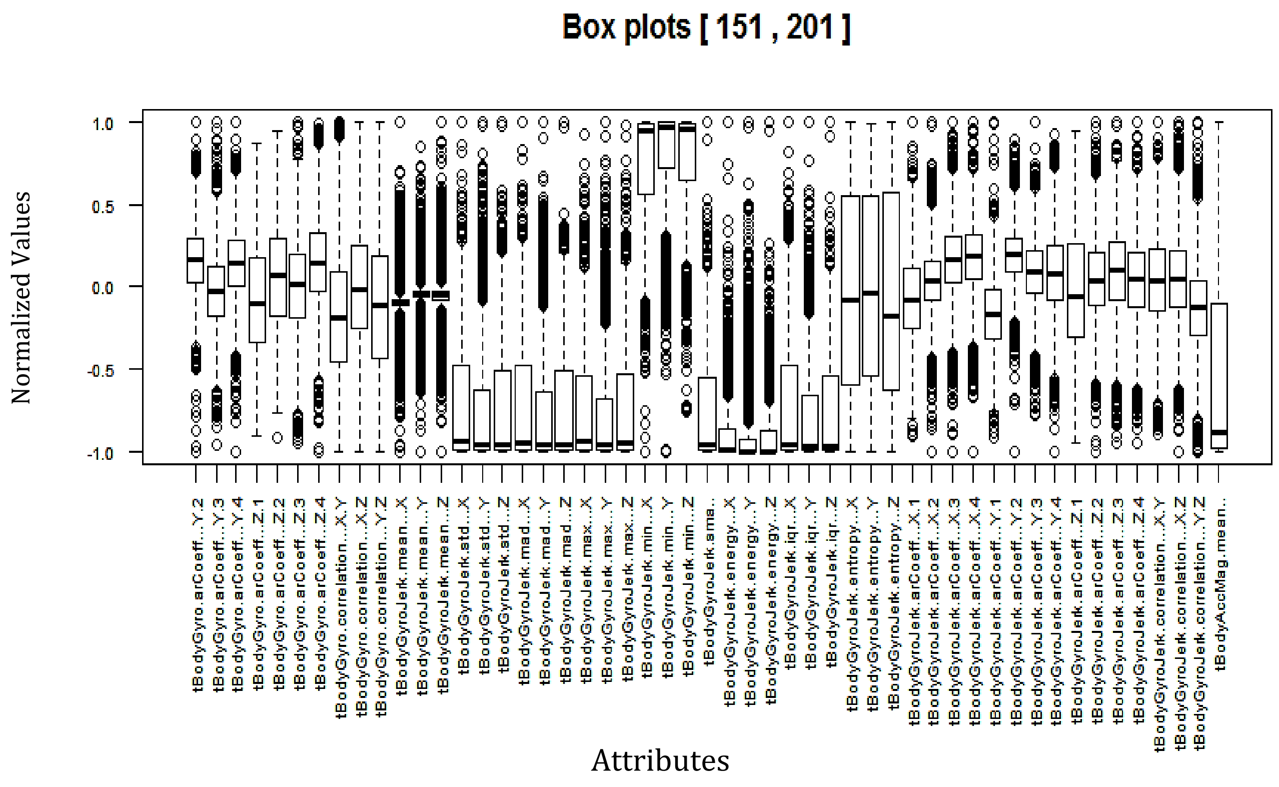


Figure 13

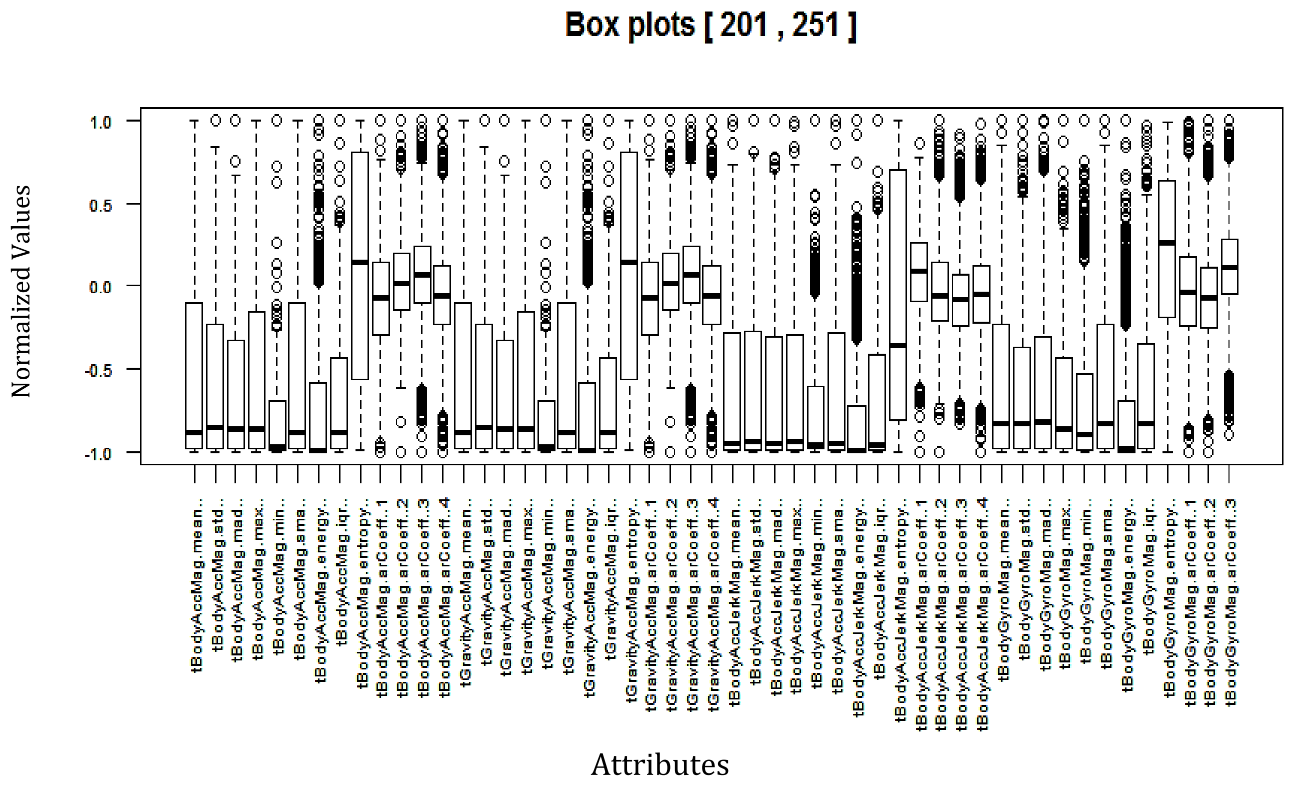


Figure 14

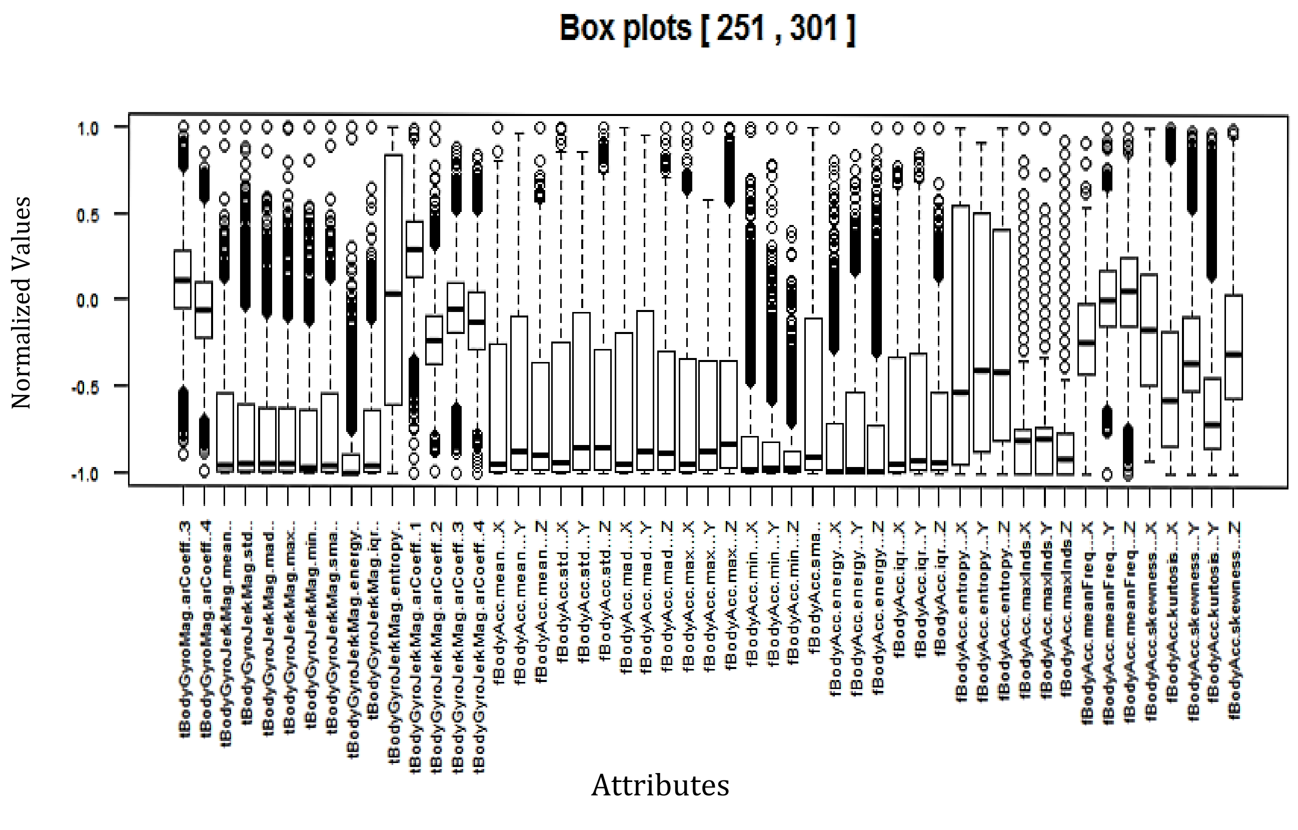


Figure 15

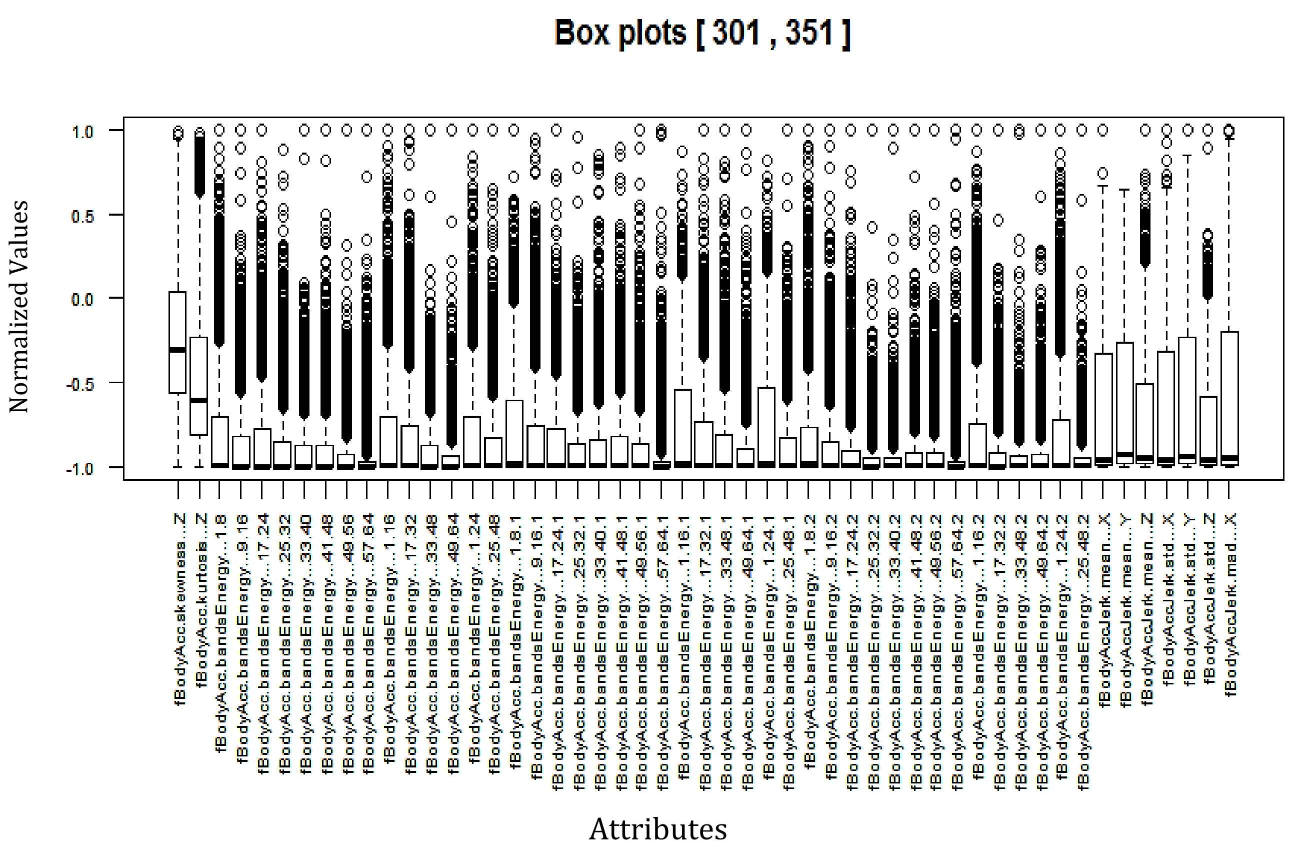


Figure 16

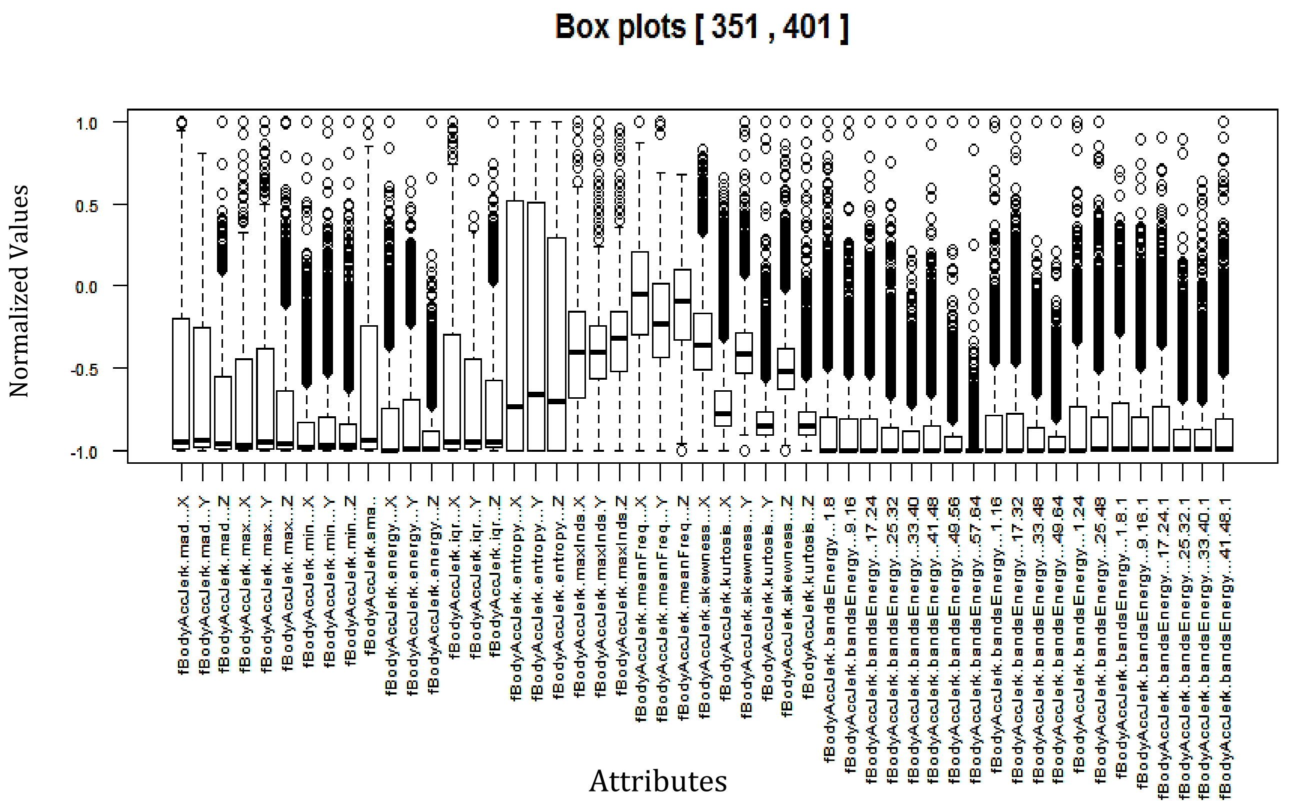


Figure 17

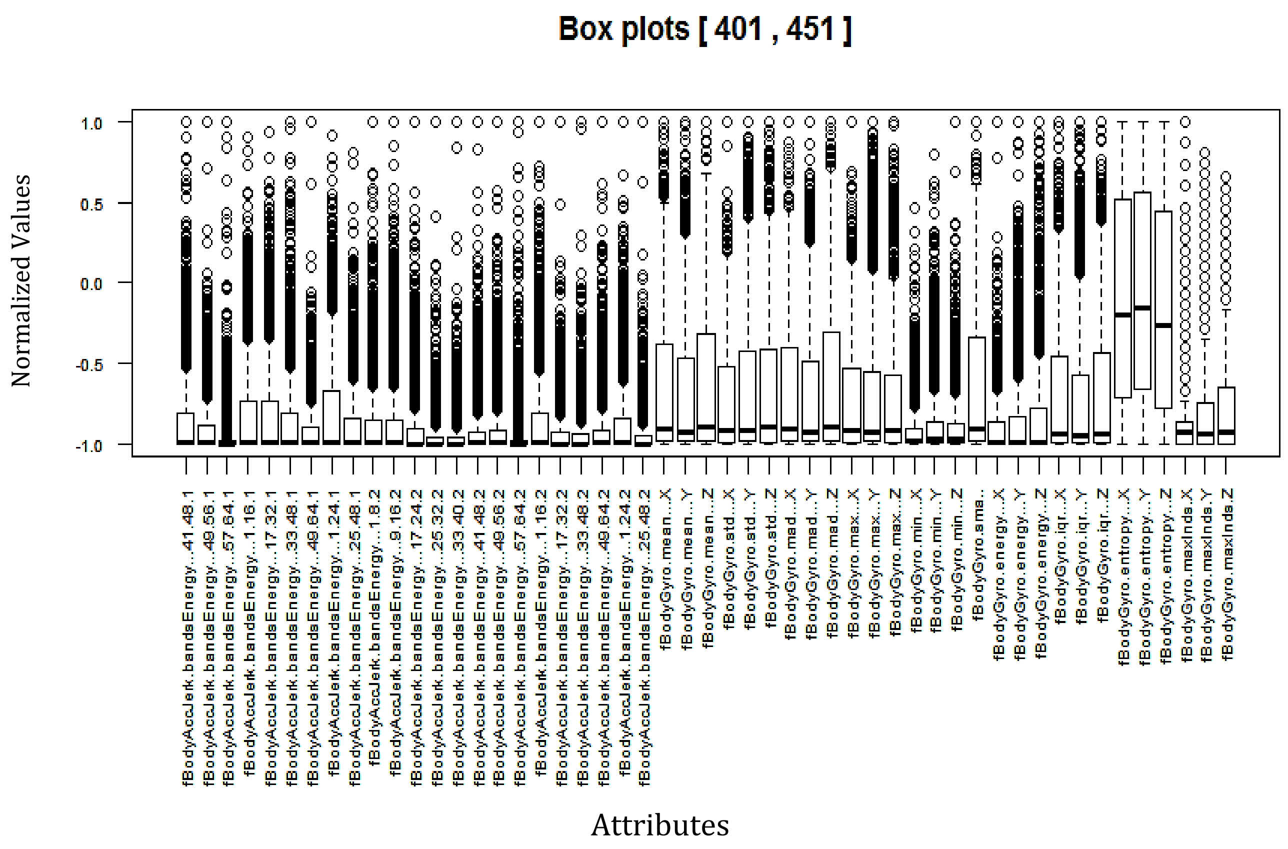


Figure 18

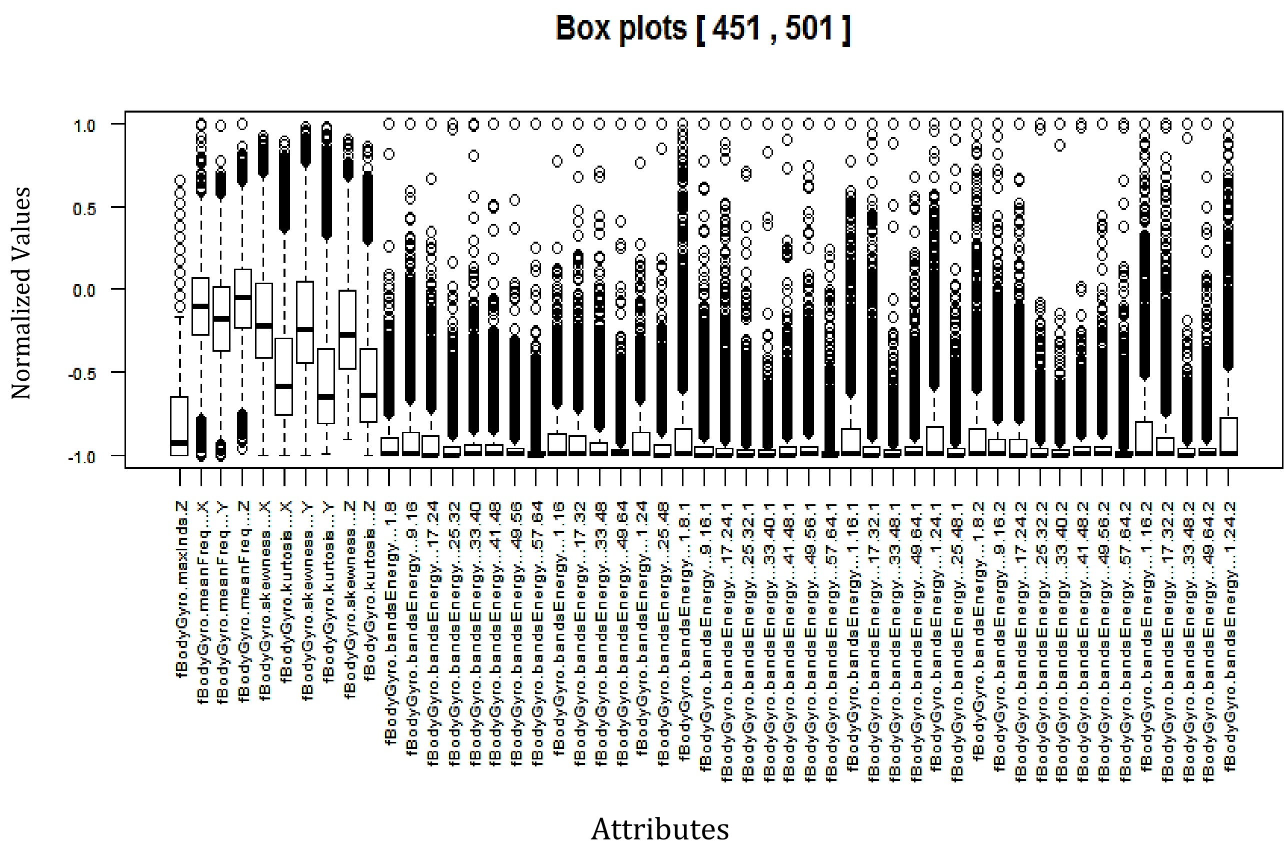


Figure 19

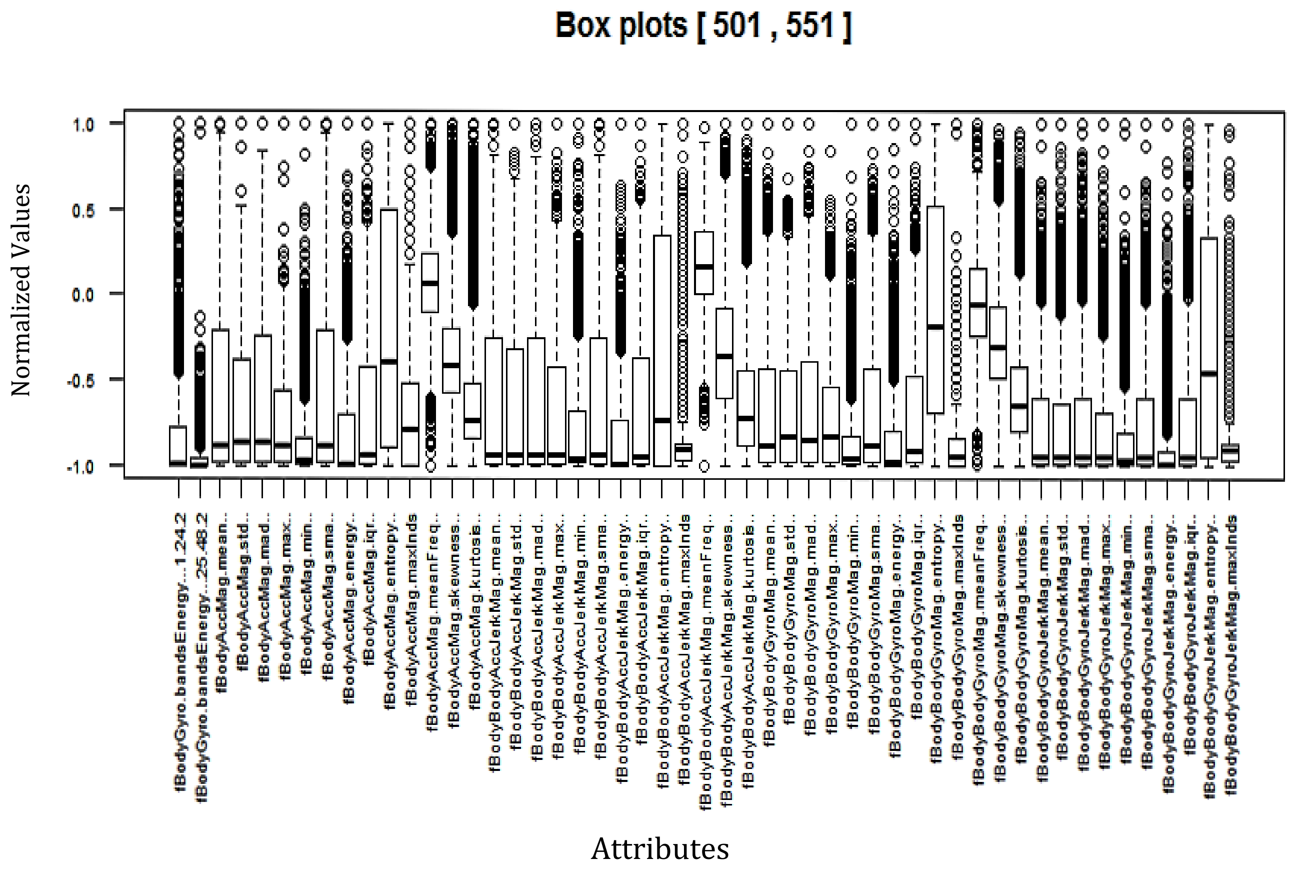
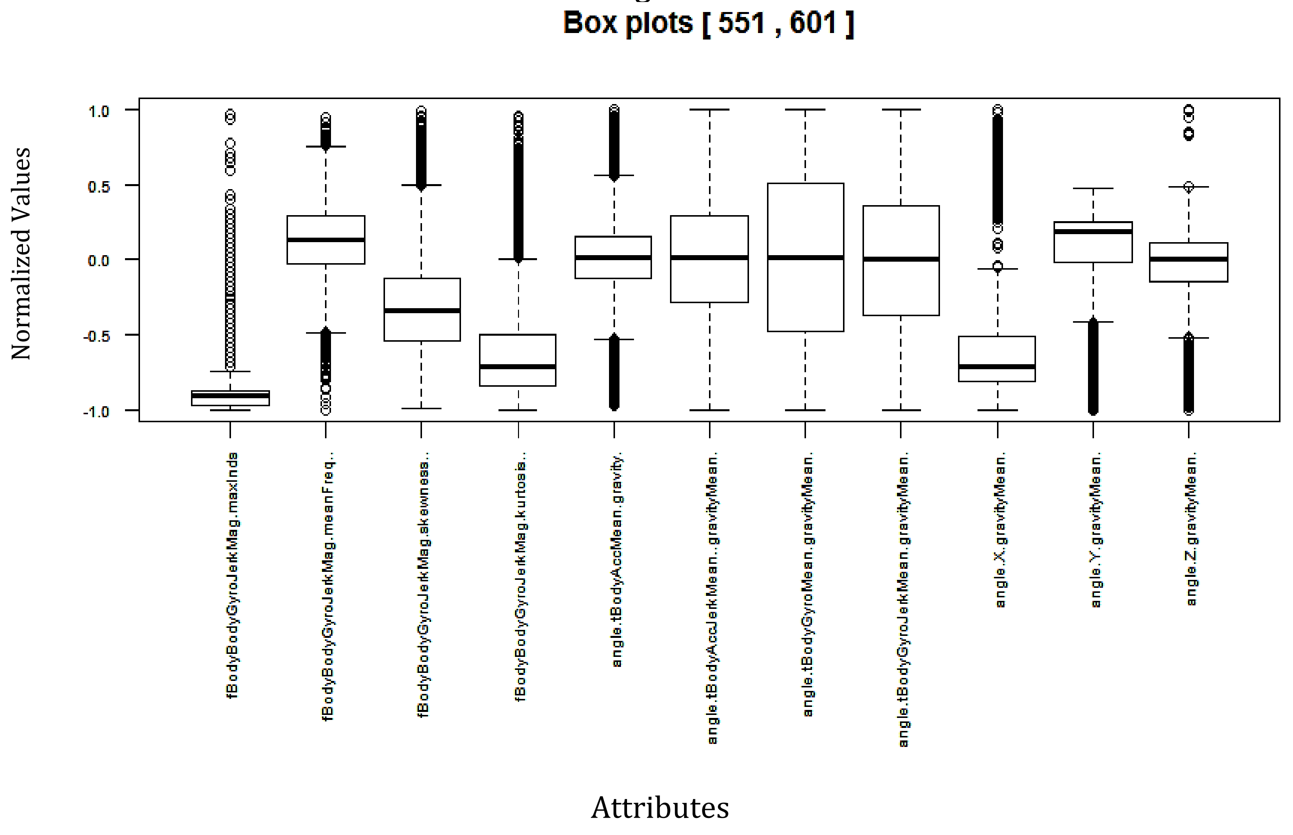


Figure 20

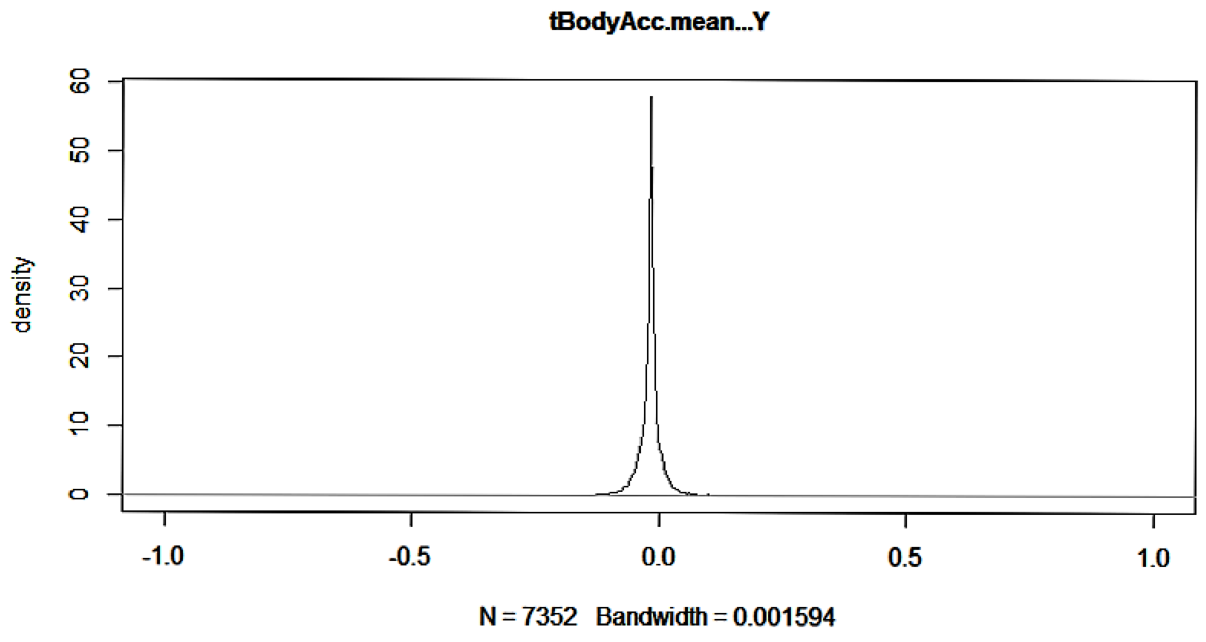


Appendix D

Skewness observations

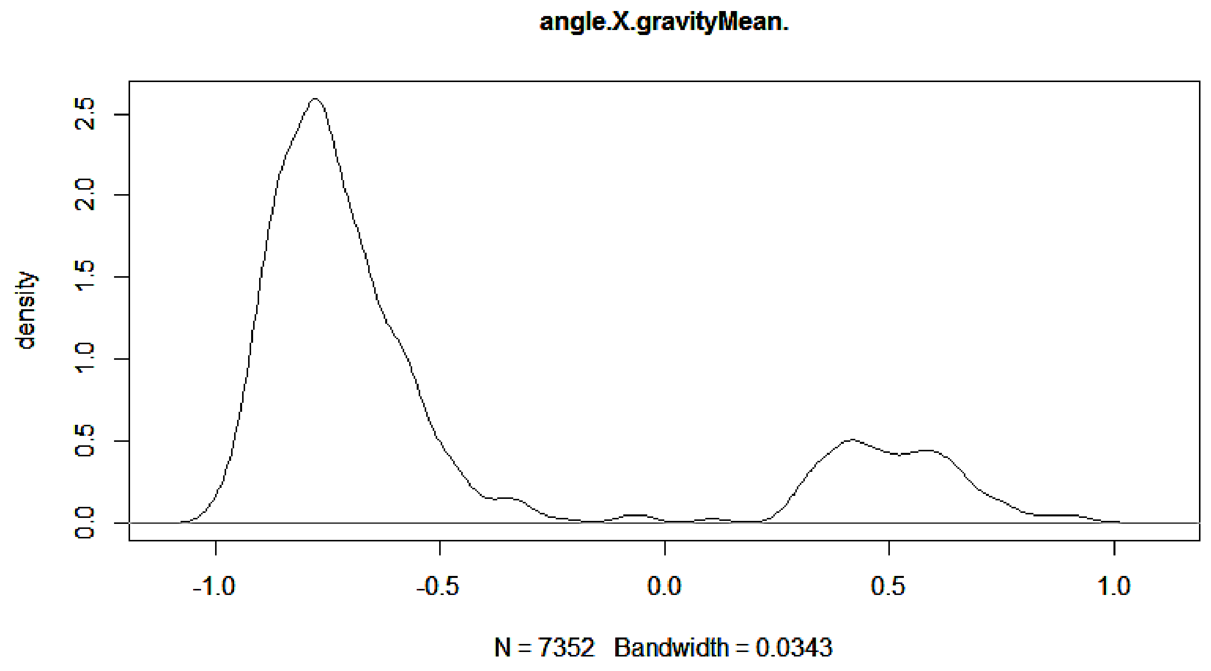
A normally distributed feature graph. The absolute value of the skewness is between 0 and 1 for these features.

Figure 21



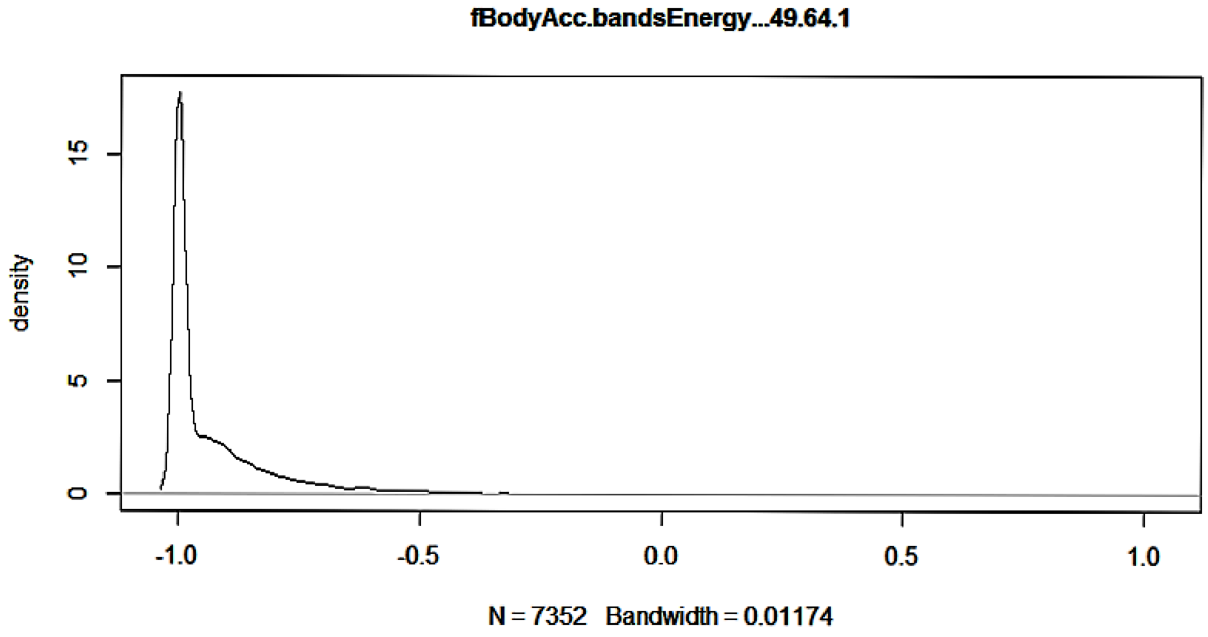
A skewed featured graph. The absolute value of the skewness is between 1 and 2 for these features.

Figure 22



A highly skewed feature graph. The absolute value of the skewness is greater than 2.

Figure 23



Appendix E

The Random Forest Feature Selection Specifics

In order to employ the random forest algorithm with 5-fold cross-validation from the caret package, the train function was used on the training data with respect to the independent variable, activity. As described by the caret package, the train function "sets up a grid of tuning parameters for a number of classification and regression routines, fits each model and calculates a resampling based performance measure."[7]

The use of 5-fold cross-validation was used in order to better generalize the results without the necessary creation of a specific validation set. The reason five folds were chosen was for algorithm run-time. Even using parallel processing when computing across the validation sets, five folds takes approximately 20 minutes to run the random forest algorithm. Therefore, five folds were chosen instead of the standard ten folds in order to speed up the runtime of the algorithm.

From here, the most important features could be determined using the varImp() function. For reproducibility, the code is provided below.

[7] Kuhn, Max. *Package 'caret'* PDF. August 6, 2015.

```
if(!("parallel" %in% rownames(install.packages()))) {install.packages("parallel",dependencies = TRUE)}
if(!("doParallel" %in% rownames(install.packages()))) {install.packages("doParallel",dependencies =
TRUE)}

library(parallel)
library(doParallel)

set.seed(12345)

#do parallel processing
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)

#run the random forest model to select the top features
#using 5-fold cross-validation and parallel processing, time = 20 minutes
rf_model<-train(activity~.,data=train,method="rf",
        trControl=trainControl(method="cv",number=5),
        prox=TRUE,allowParallel=TRUE)

stopCluster(cluster)

#print model and final model results
print(rf_model)
print(rf_model$finalModel)

#get the most important features from the random forest results
importance = varImp(rf_model,scale=FALSE)
print(importance)
```

Appendix F

Random forest algorithm implementation code

```
if(!( "randomForest" %in% rownames(install.packages())))
{install.packages("randomForest",dependencies=TRUE)}

#run random forest algoirthm with the features to determine accuracies of the attributes
set.seed(111)

fit1 = randomForest(activity ~ tBodyAcc.mean...X,
          data = train,
          importance = TRUE)

set.seed(222)

fit2 = randomForest(activity ~ tBodyAcc.mean...X + tBodyAcc.energy...X,
          data = train,
          importance = TRUE)
```

```
set.seed(333)

fit3 = randomForest(activity ~ tBodyAcc.mean...X + tBodyAcc.energy...X + tGravityAcc.mean...Y,
                data = train,
                importance = TRUE)

set.seed(444)

fit4 = randomForest(activity ~ tBodyAcc.mean...X + tBodyAcc.energy...X + tGravityAcc.mean...Y +
angle.X.gravityMean.,
                data = train,
                importance = TRUE)


set.seed(555)

fit5 = randomForest(activity ~ tBodyAcc.mean...X + tBodyAcc.energy...X + tGravityAcc.mean...Y +
angle.X.gravityMean. + tGravityAcc.min...X,
                data = train,
                importance = TRUE)

# calculate the accuracies for the rf for the features
fit1Acc = 1 - mean(predict(fit1) != train$activity)
fit2Acc = 1 - mean(predict(fit2) != train$activity)
fit3Acc = 1 - mean(predict(fit3) != train$activity)
fit4Acc = 1 - mean(predict(fit4) != train$activity)
fit5Acc = 1 - mean(predict(fit5) != train$activity)

#create array of accuracies
theAccuracies = c(fit1Acc, fit2Acc, fit3Acc, fit4Acc, fit5Acc)

#create plot of accuracies
plot(theAccuracies,
    type = "b",
    main = "Accuracy for Features Selected",
    xlab = "Number of features",
    ylab = "Accuracy")
```

Appendix G

Figure 24

Table of accuracy results

Number of Features	**Training Accuracy**	**Test Accuracy**
1	27.26%	26.98%
2	50.51%	48.23%
3	83.58%	81.61%
4	91.26%	89.03%
5	93.95%	91.14%
561	98.10%	96.12%

Appendix H

Figure 25

Table of skewness for the top five features selected for the models

Feature Name	**Skewness Value**
tGravityAcc.mean…X	-1.61
tGravityAcc.energy…X	-1.40
tGravityAcc.mean…Y	1.21
angle.X.gravityMean.	1.40
tGravityAcc.min…X	-1.61

References

"Assumptions/Limitations of Random Forest Models." Stack Exchange. Accessed October 04, 2016. http://datascience.stackexchange.com/questions/6015/assumptions-limitations-of-random-forest-models.

Kuhn, Max. "A Short Introduction to the Caret Package." August 6, 2015. Accessed October 3, 2016. https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf.

Kuhn, Max. *Package 'caret'* PDF. August 6, 2015.

Reyes-Ortiz, Jorge L., Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. *README*. Txt. November 2013.