Human Activity Recognition Using Smartphone Data

Project report by:

Nilesh Patil

Human Activity Recognition

Introduction:

A modern smartphone comes equipped with variety of sensors from motion detectors to optical calibrators. The data collected by these sensors is valuable for better aligning the applications on the phone with user's lifestyle. In this project, we have focused on using data collected from motion sensors to build a model which identifies type of activity being performed with minimal computation involved. The end goal is to create a model which can classify the activity being performed with high accuracy without sacrificing the limited computational resources available on a single phone.

Data Collection and Preparation:

We used the data provided by Human Activity Recognition research project, which built this database from the recordings of 30 subjects performing activities of daily living (ADL) while carrying a waist-mounted smartphone with embedded inertial sensors. The complete data & related papers can be accessed at: UCI ML repository page

Data was collected for 30 volunteers whose age was between 19-48 years. Each record in the data represents information about features like acceleration along x,y,z axes, velocity along a,y,z axes, 561 attributes derived from these basic measurements, identifier variable for the user & the activity being performed.

There are 6 categories of activities being performed:

1. 'standing'
2. 'sitting'
3. 'laying'
4. 'walk'
5. 'walkdown'
6. 'walkup'

The raw data has separate text files for most of the variable groups & we have used the dataset that was saved as RData file. In this dataset, a single column('subject') is used to identify a user and the last column('activity') was used to identify the activity being performed when the measurements were taken. All other attributes are available in the same column oriented data format. This is important to know, because, the values in the dataset have been normalized.

Exploratory Analysis:

High dimensionality:

The dataset contains 561 features and we started out by exploring how these are related to each other & whether there are some which can be safely ignored for our problem.

Correlation Check:

We built a correlation matrix (Appen : Fig-01) for all 561 variables in one got to identify any apparent patterns in the relationships. We see that most of these features are highly correlated with each other and it's a good decision to drop most of these highly correlated features since we can get the same information from some other feature with high correlation to a group of them.

Variance Check:

We checked our variable for zero or low variance so that they can be removed before running any analysis. Variables which do not change have low variance and 'll eventually have smaller impact on the classification model itself.

Missing value Check:

We checked for any missing values in our columns, which might lead to errors in any future analysis but didn't find any and so proceeded with the complete dataset.

Visual exploration:

We also started out with basic visual exploration of the dataset by plotting distributions for the variables for each category, but given the large number involved, we dropped the idea. Though, in general there are two distinct major groups which we can see through the distributions as shown in (Appen : Fig-02)

Methods:

The first step was to create a train & test set. We split our data into two sets in 7:3 ratios by random sampling without replacement. This ensures that our train & test sets are representative of the complete dataset. Another approach to do it would be to do this sampling for each output class. In our case, the result wasn't significantly different.

For modelling, we used the following techniques on our training set:

- SVM – Support vector machines
- Random forest (Final Model)

To determine stability of the model being used, we use OOB score calculated during model building phase as representative of the validation set & optimized our model to increase this score. For determining true performance, we used a separate test set which was not included in any of our variable selection, model training or validation phases. A high accuracy on this independent test set is proof that the model is not overfitting our training data & hence, should generalize well.

We used Random forest variable importance scores to determine the final variables to build our submission model. This process of variable selection was done iteratively & various parameters were tested. To maintain reproducibility, we set RandomState=42 at the beginning of the code so as to have uniform train/test sets & variables every time we run this code.

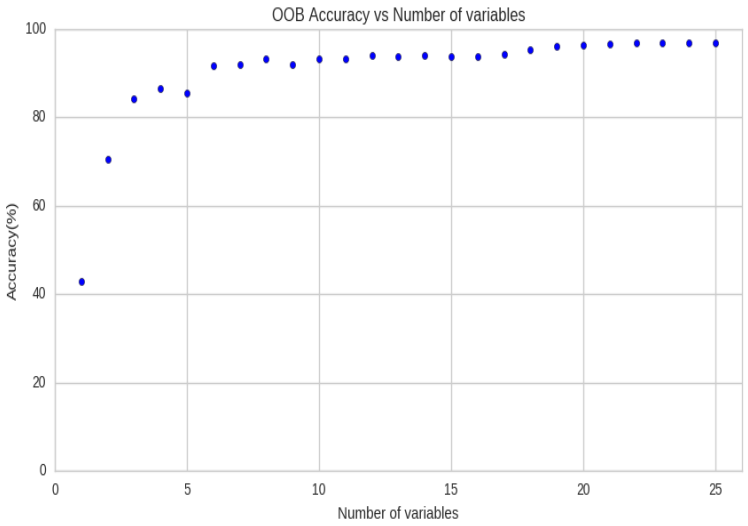
We started out with all 561 variables & reduced the total features to 5 in our final model. The focus of our process was to follow a algorithmic approach instead of a domain knowledge based model building process & hence we relied on oob score & variable importance to determine the optimal number of features, trees to be used & which features to use.

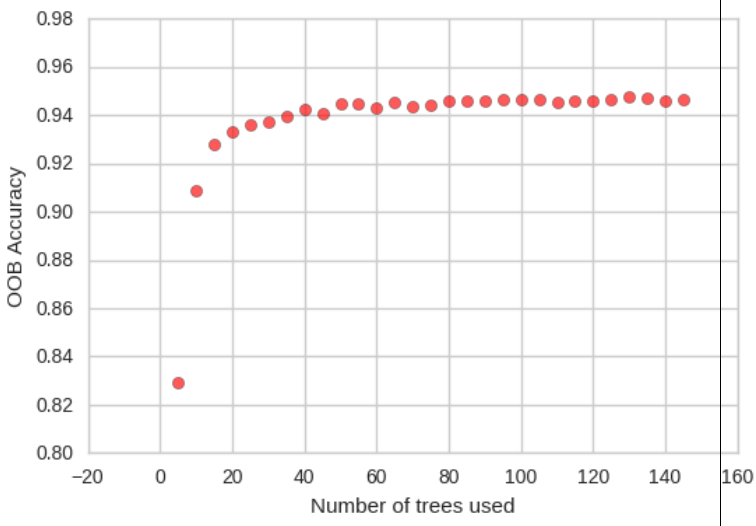
Step-by-step process:

1. Set RandomState = 42
2. Split data into two sets:
    i. train(70%)
    ii. test(30%)
3. Using training set & ALL features, build a random forest ensemble
4. From variable importance measure generated during the previous step, rank features according to their importance in differentiating between categories (Appen : Fig-03)
5. Determine optimal number of trees & variables by iterating over 0-150 trees & for 1-25 variables
6. For the final step, we use 5 of the most important measures determined in this fashion & number of trees = 50
7. Using oob score during training phase & accuracy score from the test set as final step, we freeze this model for final submission

The project is hosted here: Github Repository Link

The only, major assumption in our choice of algorithm (RandomForest) is that random forests don't overfit training set. This assumption breaks down when the training dataset is extremely biased, but in our case its relatively balanced & hence we choose it over other algorithms.





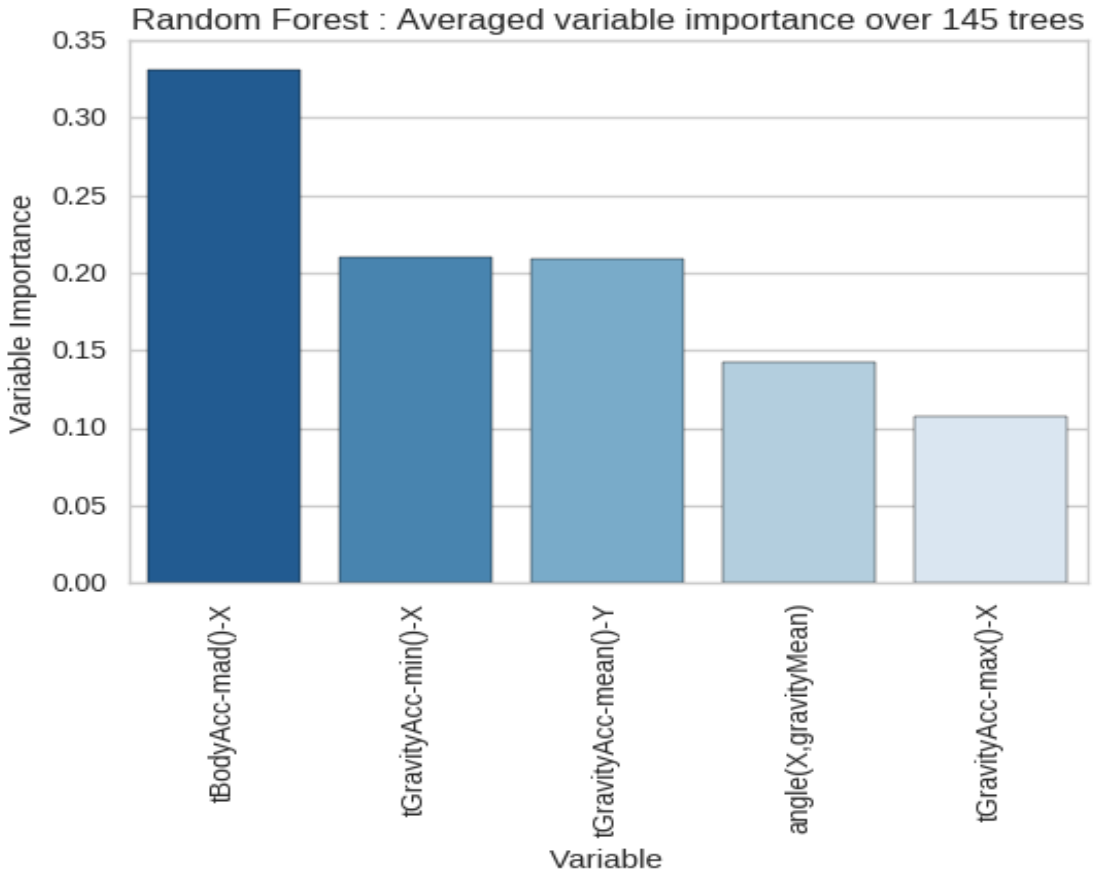
Analysis and Results:

1. Important Features:

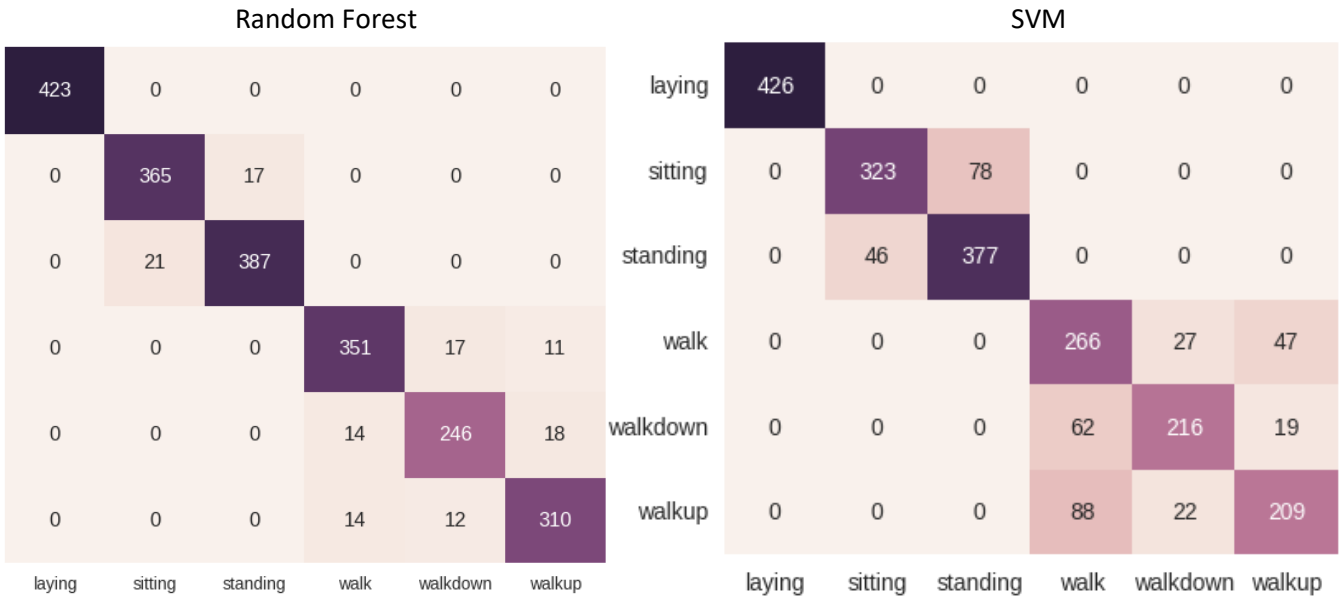
   Using the previously described feature selection, we determined that the following features were important for building our classification model:

   i. angle(X,gravityMean)
   ii. tGravityAcc-mean()-Y
   iii. tGravityAcc-min()-X
   iv. tGravityAcc-max()-X
   v. tBodyAcc-mad()-X

The final model had importance scores are as shown in the figure:



2. We used SVM & RandomForest for the final model & their accuracy scores along with confusion matrices are as shown:



Accuracy Scores:

	Random Forest	**SVM**
Train	94.50% (oob)	83.48%
Test	94.37%	82.37%

Given the high score we get on test dataset, we are confident in using RandomForest based model for deticting human activity from smartphone dataset.

From the final model, we also see that some categories are fairly straightforward to classify compared to others. We have shown this using a scatterplot matrix colored by category in (Appen : Fig-05)

Conclusions:

Overall, we relied heavily on RandomForest & SVM. It would have been nice to have been asked to try out multiple approaches as part of the project itself and include results from each & maybe delve deeper into any differences.

References:

- Random Forest:
  - https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm
  - http://scikit-learn.org/stable/modules/ensemble.html#forest
  - https://en.wikipedia.org/wiki/Random_forest
- SVM:
  - http://scikit-learn.org/stable/modules/svm.html
  - https://en.wikipedia.org/wiki/Support_vector_machine
- OOB Score:
  - https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr
  - http://scikit-learn.org/stable/auto_examples/ensemble/plot_ensemble_oob.html
- UCI-ML dataset location:
  - https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones
- Scikit-Learn:
  - http://scikit-learn.org/stable/index.html
- Github Page:
  - https://github.com/nilesh-patil/HumanActivityRecognition

Appendices:

Fig 01. Correlation Matrix between all 561 features

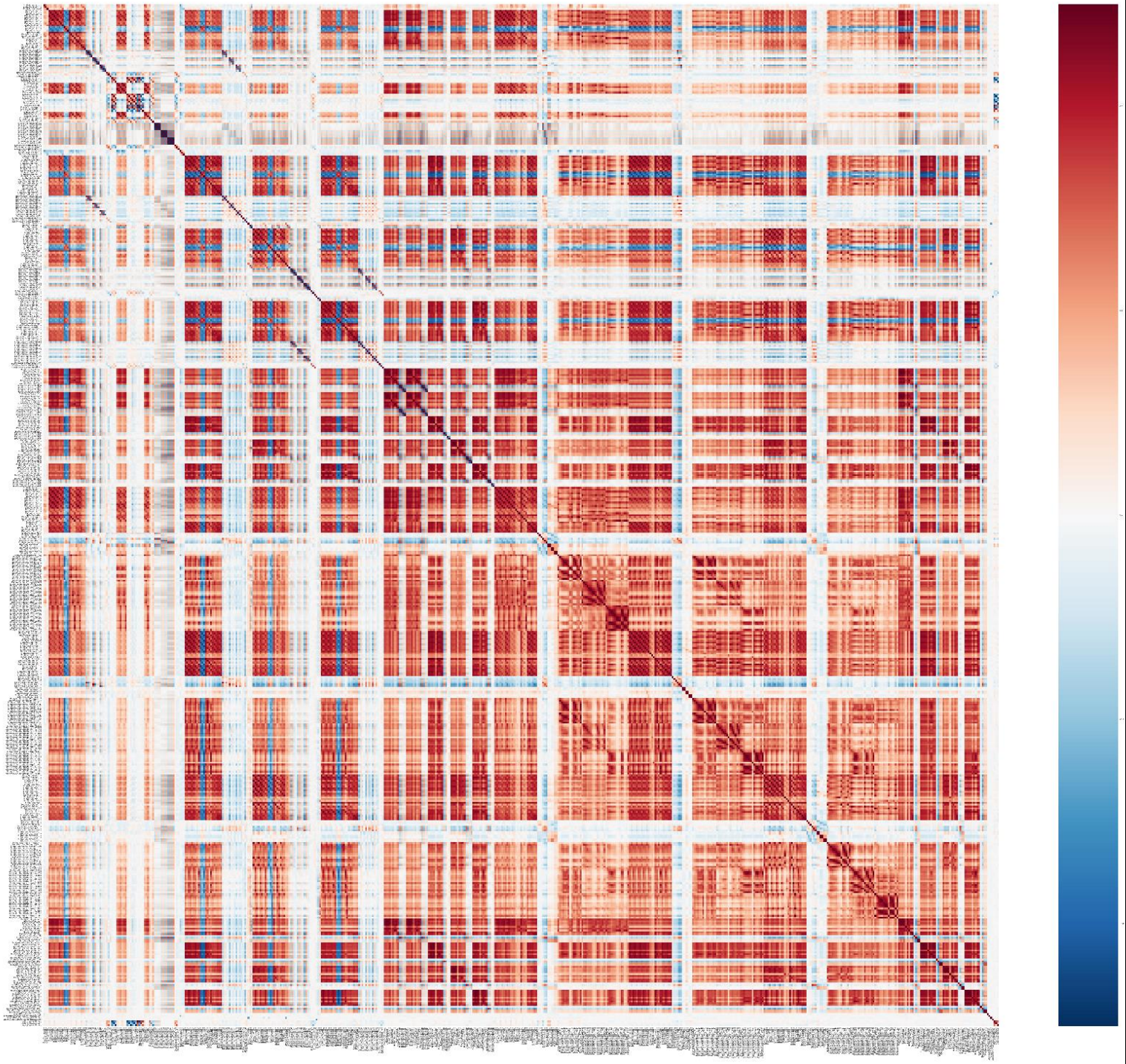


Fig 02. Distribution of tBodyAccJerk-std()-X across all 6 categories

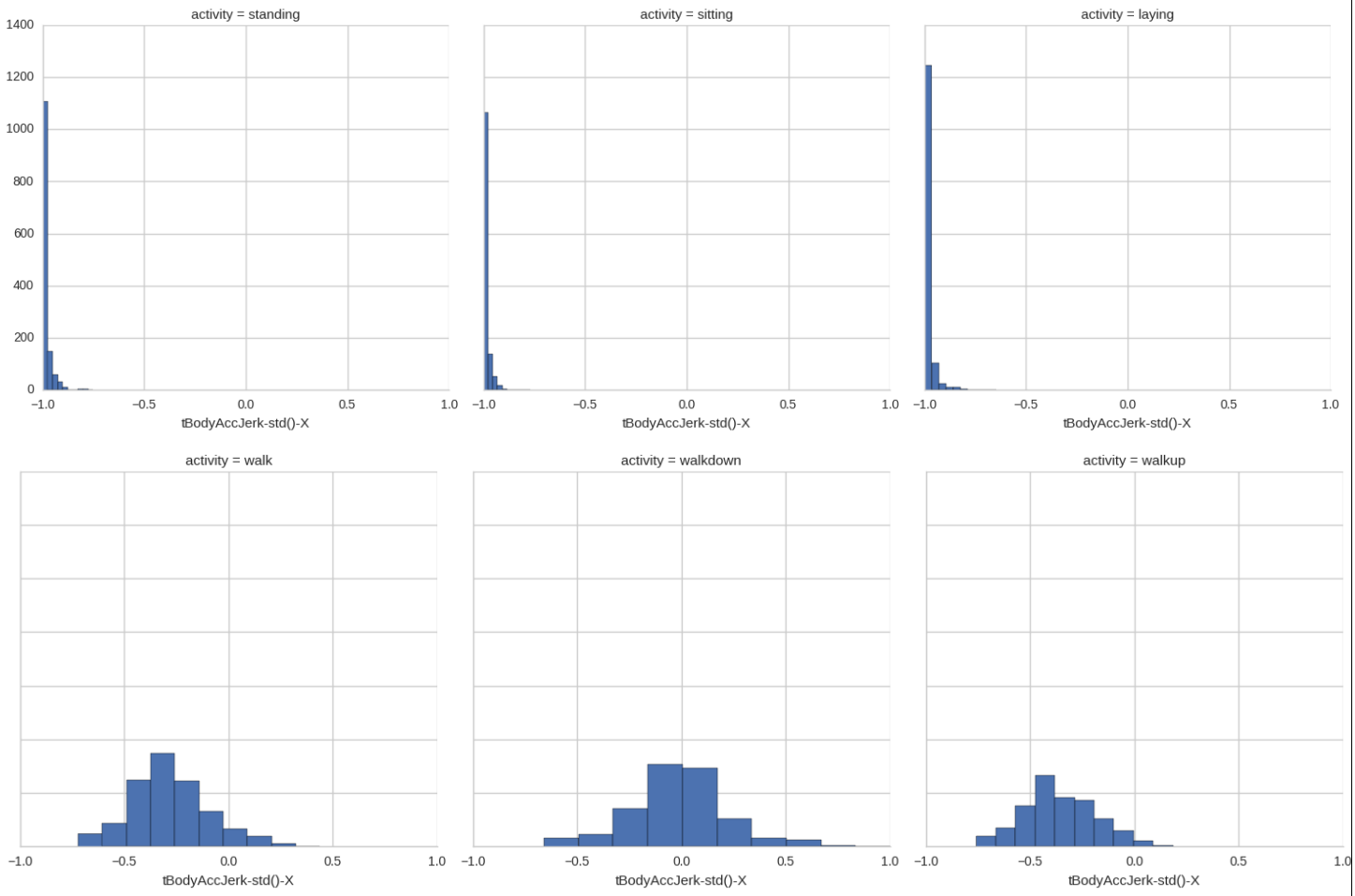


Fig 03. Variable importance: baseline model

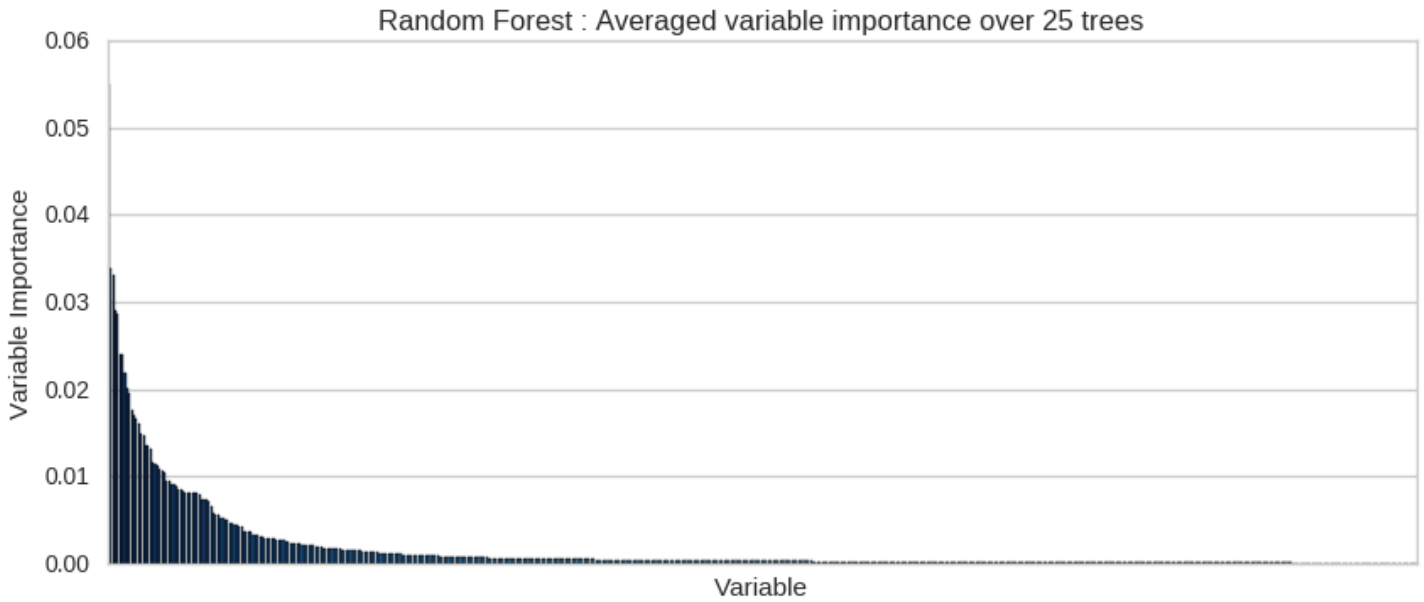


Fig 04. Distribution of tBodyAccJerk-std()-X across all 6 categories