Getting Tidy Data Codebook

Getting and Cleaning Data Codebook

This is my first take on how to make a codebook. This codebook is an enhancement on the code book from the UCI HAR Dataset. A reference of the dataset: Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine. International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012

Function parameters

- Folder: the type of data files in the folder
    1. Test
    2. Train
- Features: a list containing the output of the parseFeaturesFile function

The variables for each of the columns in the test, train, allData, and the output of forQuestion5 data frames.

- Subject: the volunteer participating in the experiment (30 people total) whose age Bracket is between 19-48 years old
- Activity: activities the person conducted while wearing a Samsung Galaxy S II
    1. Walking
    2. Walking Upstairs
    3. Walking Downstairs
    4. Sitting
    5. Standing
    6. Laying

Background info for the other variables:

Below is more background information from the UCI HAR Dataset.

Using the embedded accelerometer and gyroscope in the Samsung Galaxy S II, the experiment captured 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz. The experiments have been video-recorded to label the data manually. The obtained dataset has been randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data.

The sensor signals (accelerometer and gyroscope) were pre-processed by applying noise filters and then sampled in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window). The sensor

acceleration signal, which has gravitational and body motion components, was separated using a Butterworth low-pass filter into body acceleration and gravity. The gravitational force is assumed to have only low frequency components, therefore a filter with 0.3 Hz cutoff frequency was used. From each window, a vector of features was obtained by calculating variables from the time and frequency domain. See 'features_info.txt' for more details.

For each record it is provided:

- Triaxial acceleration from the accelerometer (total acceleration) and the estimated body acceleration.
- Triaxial Angular velocity from the gyroscope.
- A 561-feature vector with time and frequency domain variables.
  * Features are normalized and bounded within [-1,1]*

The features selected for this database come from the accelerometer and gyroscope 3-axial raw signals tAcc-XYZ and tGyro-XYZ. These time domain signals (prefix 't' to denote time) were captured at a constant rate of 50 Hz. Then they were filtered using a median filter and a 3rd order low pass Butterworth filter with a corner frequency of 20 Hz to remove noise. Similarly, the acceleration signal was then separated into body and gravity acceleration signals (tBodyAcc-XYZ and tGravityAcc-XYZ) using another low pass Butterworth filter with a corner frequency of 0.3 Hz.

Subsequently, the body linear acceleration and angular velocity were derived in time to obtain Jerk signals (tBodyAccJerk-XYZ and tBodyGyroJerk-XYZ). Also the magnitude of these three-dimensional signals were calculated using the Euclidean norm (tBodyAccMag, tGravityAccMag, tBodyAccJerkMag, tBodyGyroMag, tBodyGyroJerkMag).

Finally a Fast Fourier Transform (FFT) was applied to some of these signals producing fBodyAcc-XYZ, fBodyAccJerk-XYZ, fBodyGyro-XYZ, fBodyAccJerkMag, fBodyGyroMag, fBodyGyroJerkMag. (Note the 'f' to indicate frequency domain signals).

These signals were used to estimate variables of the feature vector for each pattern: '-XYZ' is used to denote 3-axial signals in the X, Y and Z directions.

<u>The variables in the data frames used in my scripts:</u> As the assignment was only interested in the std() (standard deviation) and mean of the data, below are the variables found within the tables

- "tBodyAcc-std()-X"
- "tBodyAcc-std()-Y"
- "tBodyAcc-std()-Z"
- "tGravityAcc-std()-X"
- "tGravityAcc-std()-Y"
- "tGravityAcc-std()-Z"
- "tBodyAccJerk-std()-X"
- "tBodyAccJerk-std()-Y"
- "tBodyAccJerk-std()-Z"
- "tBodyGyro-std()-X"
- "tBodyGyro-std()-Y"
- "tBodyGyro-std()-Z"

<u>The variables in the data frames used in my scripts cont.</u>

Getting Tidy Data Codebook

- "tBodyGyroJerk-std()-X"
- "tBodyGyroJerk-std()-Y"
- "tBodyGyroJerk-std()-Z"
- "tBodyAccMag-std()"
- “fBodyAcc-std()-X"
- "fBodyAcc-std()-Y"
- "fBodyAcc-std()-Z"
- "fBodyAccJerk-std()-X"
- "fBodyAccJerk-std()-Y"
- "fBodyAccJerk-std()-Z"
- "fBodyGyro-std()-X"
- "fBodyGyro-std()-Y"
- "fBodyGyro-std()-Z"
- "fBodyAccMag-std()"
- "fBodyBodyAccJerkMag-std()"
- "fBodyBodyGyroMag-std()"
- "fBodyBodyGyroJerkMag-std()"
- "tBodyAcc-mean()-X"
- "tBodyAcc-mean()-Y"
- "tBodyAcc-mean()-Z"
- "tGravityAcc-mean()-X" "
- "tGravityAcc-mean()-Y"
- "tGravityAcc-mean()-Z"
- "tBodyAccJerk-mean()-X"
- "tBodyAccJerk-mean()-Y"
- "tBodyAccJerk-mean()-Z"
- "tBodyGyro-mean()-X"
- "tGravityAccMag-std()"
- "tBodyAccJerkMag-std()"
- "tBodyGyroMag-std()"
- "tBodyGyroJerkMag-std()"
- "tBodyGyro-mean()-Y"
- "tBodyGyro-mean()-Z"
- "tBodyGyroJerk-mean()-X"
- "tBodyGyroJerk-mean()-Y"
- "tBodyGyroJerk-mean()-Z"
- "tBodyAccMag-mean()"
- "tGravityAccMag-mean()"
- "tBodyAccJerkMag-mean()"
- "tBodyGyroMag-mean()"
- "tBodyGyroJerkMag-mean()"
- "fBodyAcc-mean()-X"
- "fBodyAcc-mean()-Y"
- "fBodyAcc-mean()-Z"
- "fBodyAccJerk-mean()-X"
- "fBodyAccJerk-mean()-Y"
- "fBodyAccJerk-mean()-Z"
- "fBodyGyro-mean()-X"
- "fBodyGyro-mean()-Y"
- "fBodyGyro-mean()-Z"
- "fBodyAccMag-mean()"
- "fBodyBodyAccJerkMag-mean()"
- "fBodyBodyGyroMag-mean()"
- "fBodyBodyGyroJerkMag-mean(