Analysis Codebook

Yashar

August 25, 2019

Project Description

The purpose of this project is to collect and clean a data set to prepare tidy data that can be used for later analysis.

Study design and data processing

Collection of the raw data

The experiments have been carried out with a group of 30 volunteers within an age bracket of 19-48 years. Each person performed six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, we 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 reference [1]

Notes on the original (raw) data

N/A

Creating the tidy datafile

Guide to create the tidy data file

The following steps were taken to create the tidy data file:

1. download the data from the following link, and save the "UCI HAR Dataset" folder in my working directory: link
2. Reading "features.txt" file the 2nd Column of which will be used to rename xtest dataframe columns.

```
features <- read.table(file = "./UCI HAR Dataset/features.txt", header = FALSE)
```

3. Reading "X_test.txt" and "X_train.txt" file, and renaming the columns using a vector extracted from the above "features" variable. This should satisfy part 4 of the assignment.

```
xtest <- read.table(file = "./UCI HAR Dataset/test/X_test.txt", header = FALSE, col.names = features$V2
xtrain <- read.table(file = "./UCI HAR Dataset/train/X_train.txt", header = FALSE, col.names = features
```

4. Reading "y_test.txt" file and "y_train.txt" file, and renaming the column to "activity.class".

```
ytest <- read.table(file = "./UCI HAR Dataset/test/y_test.txt", header = FALSE, col.names = "activity.cl
ytrain <- read.table(file = "./UCI HAR Dataset/train/y_train.txt", header = FALSE, col.names = "activity
```

5. Reading "subject_test.txt" file, and renaming the column to "subject.id".

```
subjecttest <- read.table(file = "./UCI HAR Dataset/test/subject_test.txt", header = FALSE, col.names =
subjecttrain <- read.table(file = "./UCI HAR Dataset/train/subject_train.txt", header = FALSE, col.name
```

Cleaning of the data

A Short, high-level description of cleaning process is presented in this section.

6. Merging 3 sets of entities - subjects, activities, and readings.Then adding a new column named "origin" with values of "test" or "train" to specify the origin of data. dplyr package is also called from the library to use "tbl_df" function.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
merged.test.data <- tbl_df(cbind(subjecttest, ytest, xtest))
merged.test.data$origin <- "test"
merged.train.data <- tbl_df(cbind(subjecttrain, ytrain, xtrain))
merged.train.data$origin <- "train"
```

7. Merging the training and the test sets to create one data set and fulfill 1st part of the assignment.

```
merged.data.set <- rbind(merged.train.data, merged.test.data)
```

8. Identifying columns labels that contain "Mean", "mean", "Std" , or "std". This will be used to extract only the measurements on the mean and standard deviation for each measurement, and complete 2nd part of assignment. "select" function from dplyr package is used to extract desired columns.

```
selected_columns <- grep("[Mm]ean|[Ss]td", names(merged.data.set), value = TRUE)
extracted_dataset <- select(merged.data.set, subject.id, activity.class, origin, selected_columns)
```

9. Reading "activity_labels.txt" file. The descriptive activity names in this file will be used to name the activities in the merged data set, and fulfill part 3 of the assignment.

```
activity_labels <- read.table(file = "./UCI HAR Dataset/activity_labels.txt", header = FALSE, col.names
renamed.activity <- tbl_df(merge(x = activity_labels, y = merged.data.set, by.x = "activity_id", by.y =
```

10. Creating a tidy data set with the average of each variable for each activity and each subject. "group_by" and "summarise_all" functions from dplyr package were used to perform this operation and complete section 5 of the assignment. Then the outcome was written to a file "tidydata.txt".

```
tidy.data <- renamed.activity %>% group_by(activity_description, subject.id) %>% summarise_all(mean)
```

```
## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA
```

```
## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA
```

```
## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
```

```
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA

## Warning in mean.default(origin): argument is not numeric or logical:
## returning NA
```

```r
write.table(tidy.data, file = "./tidydata.txt", row.names = FALSE)
```

Description of the variables in the tidydata.txt file

General description of the file: - Dimensions of the dataset: 180 observations of 565 variables - Summary of the data: Running the structure function "str" to obtain a summary of the tidy data:

```
str(tidy.data)
```

```
## Classes 'grouped_df', 'tbl_df', 'tbl' and 'data.frame':  180 obs. of  565 variables:
##  $ activity_description                : Factor w/ 6 levels "LAYING","SITTING",..: 1 1 1 1 1 1 1 1 1
##  $ subject.id                          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ activity_id                         : num  6 6 6 6 6 6 6 6 6 6 ...
##  $ tBodyAcc.mean...X                   : num  0.222 0.281 0.276 0.264 0.278 ...
##  $ tBodyAcc.mean...Y                   : num  -0.0405 -0.0182 -0.019 -0.015 -0.0183 ...
##  $ tBodyAcc.mean...Z                   : num  -0.113 -0.107 -0.101 -0.111 -0.108 ...
##  $ tBodyAcc.std...X                    : num  -0.928 -0.974 -0.983 -0.954 -0.966 ...
##  $ tBodyAcc.std...Y                    : num  -0.837 -0.98 -0.962 -0.942 -0.969 ...
##  $ tBodyAcc.std...Z                    : num  -0.826 -0.984 -0.964 -0.963 -0.969 ...
##  $ tBodyAcc.mad...X                    : num  -0.932 -0.977 -0.985 -0.958 -0.969 ...
##  $ tBodyAcc.mad...Y                    : num  -0.841 -0.981 -0.965 -0.946 -0.972 ...
##  $ tBodyAcc.mad...Z                    : num  -0.822 -0.985 -0.966 -0.963 -0.967 ...
##  $ tBodyAcc.max...X                    : num  -0.906 -0.919 -0.92 -0.913 -0.917 ...
##  $ tBodyAcc.max...Y                    : num  -0.502 -0.563 -0.55 -0.525 -0.551 ...
##  $ tBodyAcc.max...Z                    : num  -0.703 -0.811 -0.793 -0.796 -0.804 ...
##  $ tBodyAcc.min...X                    : num  0.743 0.821 0.833 0.795 0.81 ...
##  $ tBodyAcc.min...Y                    : num  0.585 0.684 0.67 0.666 0.677 ...
##  $ tBodyAcc.min...Z                    : num  0.758 0.839 0.828 0.826 0.832 ...
##  $ tBodyAcc.sma..                      : num  -0.842 -0.978 -0.971 -0.954 -0.967 ...
##  $ tBodyAcc.energy...X                 : num  -0.984 -0.999 -1 -0.992 -0.997 ...
##  $ tBodyAcc.energy...Y                 : num  -0.948 -1 -0.999 -0.997 -0.999 ...
##  $ tBodyAcc.energy...Z                 : num  -0.905 -0.999 -0.997 -0.997 -0.998 ...
##  $ tBodyAcc.iqr...X                    : num  -0.94 -0.98 -0.988 -0.969 -0.975 ...
##  $ tBodyAcc.iqr...Y                    : num  -0.877 -0.985 -0.975 -0.963 -0.978 ...
##  $ tBodyAcc.iqr...Z                    : num  -0.823 -0.985 -0.971 -0.962 -0.965 ...
##  $ tBodyAcc.entropy...X                : num  -0.372 -0.365 -0.482 -0.347 -0.377 ...
##  $ tBodyAcc.entropy...Y                : num  -0.491 -0.649 -0.569 -0.515 -0.556 ...
##  $ tBodyAcc.entropy...Z                : num  -0.402 -0.532 -0.405 -0.491 -0.459 ...
##  $ tBodyAcc.arCoeff...X.1              : num  0.043 0.0543 0.0474 0.0148 0.1197 ...
##  $ tBodyAcc.arCoeff...X.2              : num  0.00523 -0.03423 0.03616 0.0086 -0.06574 ...
##  $ tBodyAcc.arCoeff...X.3              : num  -0.0323 -0.011 -0.0879 -0.0675 -0.0584 ...
##  $ tBodyAcc.arCoeff...X.4              : num  0.1466 0.0753 0.1488 0.0104 0.1536 ...
##  $ tBodyAcc.arCoeff...Y.1              : num  0.176 0.299 0.168 0.257 0.347 ...
##  $ tBodyAcc.arCoeff...Y.2              : num  -0.1043 -0.1714 -0.0546 -0.0647 -0.0622 ...
##  $ tBodyAcc.arCoeff...Y.3              : num  0.184 0.236 0.149 0.196 0.165 ...
##  $ tBodyAcc.arCoeff...Y.4              : num  0.0105 -0.0245 0.0561 -0.0532 0.0156 ...
##  $ tBodyAcc.arCoeff...Z.1              : num  0.209 0.364 0.218 0.209 0.195 ...
##  $ tBodyAcc.arCoeff...Z.2              : num  -0.1219 -0.1591 -0.1243 -0.0709 -0.067 ...
##  $ tBodyAcc.arCoeff...Z.3              : num  0.0905 0.1761 0.159 0.0498 0.0469 ...
##  $ tBodyAcc.arCoeff...Z.4              : num  0.00239 -0.12441 -0.12485 -0.10978 -0.14461 ...
##  $ tBodyAcc.correlation...X.Y          : num  -0.040891 0.039884 -0.057748 -0.078547 0.000457 ...
##  $ tBodyAcc.correlation...X.Z          : num  -0.00933 -0.19979 -0.19859 -0.42405 0.0665 ...
##  $ tBodyAcc.correlation...Y.Z          : num  -0.0234 -0.2485 -0.3761 -0.0481 -0.0728 ...
##  $ tGravityAcc.mean...X                : num  -0.249 -0.51 -0.242 -0.421 -0.483 ...
##  $ tGravityAcc.mean...Y                : num  0.706 0.753 0.837 0.915 0.955 ...
##  $ tGravityAcc.mean...Z                : num  0.446 0.647 0.489 0.342 0.264 ...
##  $ tGravityAcc.std...X                 : num  -0.897 -0.959 -0.983 -0.921 -0.946 ...
##  $ tGravityAcc.std...Y                 : num  -0.908 -0.988 -0.981 -0.97 -0.986 ...
##  $ tGravityAcc.std...Z                 : num  -0.852 -0.984 -0.965 -0.976 -0.977 ...
##  $ tGravityAcc.mad...X                 : num  -0.899 -0.962 -0.983 -0.925 -0.95 ...
##  $ tGravityAcc.mad...Y                 : num  -0.91 -0.989 -0.982 -0.972 -0.987 ...
```

```
##  $ tGravityAcc.mad...Z                : num  -0.855 -0.985 -0.967 -0.977 -0.977 ...
##  $ tGravityAcc.max...X                : num  -0.28 -0.557 -0.303 -0.448 -0.523 ...
##  $ tGravityAcc.max...Y                : num  0.685 0.708 0.794 0.869 0.906 ...
##  $ tGravityAcc.max...Z                : num  0.469 0.636 0.484 0.335 0.257 ...
##  $ tGravityAcc.min...X                : num  -0.236 -0.469 -0.201 -0.388 -0.446 ...
##  $ tGravityAcc.min...Y                : num  0.692 0.764 0.846 0.915 0.962 ...
##  $ tGravityAcc.min...Z                : num  0.414 0.645 0.48 0.337 0.26 ...
##  $ tGravityAcc.sma..                  : num  0.248 0.271 0.293 -0.228 -0.374 ...
##  $ tGravityAcc.energy...X             : num  -0.907 -0.98 -0.947 -0.994 -0.998 ...
##  $ tGravityAcc.energy...Y             : num  0.165 0.147 0.414 0.684 0.827 ...
##  $ tGravityAcc.energy...Z             : num  -0.32 -0.184 -0.527 -0.742 -0.865 ...
##  $ tGravityAcc.iqr...X                : num  -0.906 -0.968 -0.985 -0.939 -0.961 ...
##  $ tGravityAcc.iqr...Y                : num  -0.916 -0.99 -0.985 -0.978 -0.989 ...
##  $ tGravityAcc.iqr...Z                : num  -0.863 -0.985 -0.973 -0.979 -0.978 ...
##  $ tGravityAcc.entropy...X            : num  -0.518 -0.809 -0.658 -0.609 -0.823 ...
##  $ tGravityAcc.entropy...Y            : num  -0.554 -0.804 -0.697 -0.711 -0.797 ...
##  $ tGravityAcc.entropy...Z            : num  -0.58 -0.688 -0.542 -0.664 -0.592 ...
##  $ tGravityAcc.arCoeff...X.1          : num  -0.594 -0.584 -0.506 -0.591 -0.599 ...
##  $ tGravityAcc.arCoeff...X.2          : num  0.609 0.592 0.525 0.595 0.611 ...
##  $ tGravityAcc.arCoeff...X.3          : num  -0.625 -0.6 -0.545 -0.599 -0.622 ...
##  $ tGravityAcc.arCoeff...X.4          : num  0.64 0.608 0.566 0.602 0.634 ...
##  $ tGravityAcc.arCoeff...Y.1          : num  -0.3854 -0.2087 -0.2187 -0.058 0.0841 ...
##  $ tGravityAcc.arCoeff...Y.2          : num  0.35032 0.16095 0.17961 0.00485 -0.13678 ...
##  $ tGravityAcc.arCoeff...Y.3          : num  -0.3556 -0.1651 -0.1923 -0.0133 0.1181 ...
##  $ tGravityAcc.arCoeff...Y.4          : num  0.3783 0.1917 0.2271 0.0481 -0.0692 ...
##  $ tGravityAcc.arCoeff...Z.1          : num  -0.542 -0.272 -0.426 -0.355 -0.35 ...
##  $ tGravityAcc.arCoeff...Z.2          : num  0.557 0.29 0.443 0.362 0.356 ...
##  $ tGravityAcc.arCoeff...Z.3          : num  -0.571 -0.308 -0.46 -0.369 -0.361 ...
##  $ tGravityAcc.arCoeff...Z.4          : num  0.583 0.324 0.474 0.373 0.363 ...
##  $ tGravityAcc.correlation...X.Y      : num  -0.0846 0.2156 0.1966 -0.049 -0.2782 ...
##  $ tGravityAcc.correlation...X.Z      : num  -0.0781 -0.3082 -0.4267 -0.5239 0.0457 ...
##  $ tGravityAcc.correlation...Y.Z      : num  -0.209 -0.563 -0.799 -0.155 -0.166 ...
##  $ tBodyAccJerk.mean...X              : num  0.0811 0.0826 0.077 0.0934 0.0848 ...
##  $ tBodyAccJerk.mean...Y              : num  0.00384 0.01225 0.0138 0.00693 0.00747 ...
##  $ tBodyAccJerk.mean...Z              : num  0.01083 -0.0018 -0.00436 -0.00641 -0.00304 ...
##  $ tBodyAccJerk.std...X               : num  -0.958 -0.986 -0.981 -0.978 -0.983 ...
##  $ tBodyAccJerk.std...Y               : num  -0.924 -0.983 -0.969 -0.942 -0.965 ...
##  $ tBodyAccJerk.std...Z               : num  -0.955 -0.988 -0.982 -0.979 -0.985 ...
##  $ tBodyAccJerk.mad...X               : num  -0.964 -0.987 -0.982 -0.98 -0.984 ...
##  $ tBodyAccJerk.mad...Y               : num  -0.934 -0.982 -0.968 -0.944 -0.963 ...
##  $ tBodyAccJerk.mad...Z               : num  -0.959 -0.988 -0.981 -0.978 -0.985 ...
##  $ tBodyAccJerk.max...X               : num  -0.957 -0.983 -0.979 -0.978 -0.98 ...
##  $ tBodyAccJerk.max...Y               : num  -0.944 -0.988 -0.975 -0.957 -0.973 ...
##  $ tBodyAccJerk.max...Z               : num  -0.958 -0.989 -0.982 -0.981 -0.986 ...
##  $ tBodyAccJerk.min...X               : num  0.948 0.982 0.975 0.976 0.981 ...
##  $ tBodyAccJerk.min...Y               : num  0.927 0.985 0.972 0.943 0.972 ...
##  $ tBodyAccJerk.min...Z               : num  0.93 0.985 0.977 0.973 0.982 ...
##  $ tBodyAccJerk.sma..                 : num  -0.954 -0.988 -0.979 -0.97 -0.98 ...
##   [list output truncated]
##  - attr(*, "groups")=Classes 'tbl_df', 'tbl' and 'data.frame':   6 obs. of  2 variables:
##   ..$ activity_description: Factor w/ 6 levels "LAYING","SITTING",..: 1 2 3 4 5 6
##   ..$ .rows               :List of 6
##   .. ..$ : int  1 2 3 4 5 6 7 8 9 10 ...
##   .. ..$ : int  31 32 33 34 35 36 37 38 39 40 ...
```

```
##   .. ..$ : int  61 62 63 64 65 66 67 68 69 70 ...
##   .. ..$ : int  91 92 93 94 95 96 97 98 99 100 ...
##   .. ..$ : int  121 122 123 124 125 126 127 128 129 130 ...
##   .. ..$ : int  151 152 153 154 155 156 157 158 159 160 ...
##   ..- attr(*, ".drop")= logi TRUE
```

- Variables present in the dataset: The following information is provided for each record in the dataset:
  - 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.
  - Record activity label:
    - activity_description: A factor variable consisted of 6 levels:
    1. WALKING
    2. WALKING_UPSTAIRS
    3. WALKING_DOWNSTAIRS
    4. SITTING
    5. STANDING
    6. LAYING
    - activity_id: The ID number coresponding to each of the abovementioned activities.
  - subject.id: An identifier of the subject who carried out the experiment.
  - origin: : The last column of the dataframe with values of "test" or "train" to specify the origin of data.

Token Dictionary & Codebook

The following token dictionary describes all the abbreviations based on information gleaned from the files in the assignment. in all of the measurement variables, the text tokens have the following meanings:

Token	Description
t prefix	Time domain signal.
f prefix	Frequency domain signal taken as a Fast Fourier Transform of the time-based signals.
BodyAcc	Body acceleration signal.
GravityAcc	Gravity acceleration signal.
BodyAccJerk	Jerk signal: body linear acceleration derived with respect to time.
BodyGyro	Body gyroscope signal.
BodyGyroJerk	Jerk signal: body angular velocity derived with respect to time.
BodyAccMag	Magnitude of body acceleration signal.
GravityAccMag	Magnitude of gravity acceleration signal.
BodyAccJerkMag	Magnitude of Jerk signal: body linear acceleration derived with respect to time.
BodyGyroMag	Magnitude of body gyroscope signal.
BodyGyroJerkMag	Magnitude of Jerk signal: body angular velocity derived with respect to time.
mean	Mean value.
std	Standard deviation.
mad	Median absolute deviation.
max	Largest value in array.
min	Smallest value in array.
sma	Signal magnitude area.
energy	Energy measure. Sum of the squares divided by the number of values.

Token	Description
iqr	Interquartile range.
entropy	Signal entropy.
arCoeff	Autorregresion coefficients with Burg order equal to 4.
correlation	correlation coefficient between two signals.
maxInds	index of the frequency component with largest magnitude.
meanFreq	Weighted average of the frequency components to obtain a mean frequency.
skewness	skewness of the frequency domain signal.
kurtosis	kurtosis of the frequency domain signal.
bandsEnergy	Energy of a frequency interval within the 64 bins of the FFT of each window.
angle	Angle between to vectors.

The tokens from this token dictionary were later used to build the variable labels/description in the tidy data set.

Some information on the variable including: - Class of the variable - Unique values/levels of the variable - Unit of measurement (if no unit of measurement list this as well) - In case names follow some schema, describe how entries were constructed (for example time-body-gyroscope-z has 4 levels of descriptors. Describe these 4 levels).

(you can easily use Rcode for this, just load the dataset and provide the information directly form the tidy data file)

Sources

[1] 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