**CodeBook.md**

**Terry L. Jones**

**Monday, October 8, 2018**

**Getting and Cleaning Data Course Project**

The purpose of this project is to demonstrate your ability to collect, work with, and clean a data set. The goals is to prepare tidy data that can be used for later analysis. You will be graded by your peers on a series of yes/no questions related to the project. You will be required to submit:

1. A tidy data set as described below
2. A link to a Github repository with your script for performing the analysis, and
3. A code book that describes the variables, the data, and any transformations or work that you performed to clean up the data called CodeBook.md. You should also include a README.md in the repo with your scripts. This repo explains how all of the scripts work and how they are connected.

One of the most exciting areas in all of data science right now is wearable computing – see for example [this article](http://www.insideactivitytracking.com/data-science-activity-tracking-and-the-battle-for-the-worlds-top-sports-brand/). Companies like Fitbit, Nike, and Jawbone Up are racing to develop the most advanced algorithms to attract new users. The data linked to from the course website represent data collected from the accelerometers from the Samsung Galaxy S smartphone. A full description is available at the site where the data was obtained:

<http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones>

Here are the data for the project:

<https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip>

You should create one R script called **run\_analysis.R** that does the following:

1. Merges the training and the test sets to create one data set.
2. Extracts only the measurements on the mean and standard deviation for each measurement.
3. Uses descriptive activity names to name the activities in the data set.
4. Appropriately labels the data set with descriptive variable names.
5. From the data set in Step 4, creates a second, independent tidy data set with the average of each variable for each activity and each subject.

**Description of the Data**

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.

Next, 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).

Lastly, 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).

**Description of Abbreviations of Measurements**

1. Acc = accelerometer measurement
2. Body = related to body movement
3. Gravity – acceleration of gravity
4. Gyro – gyroscopic measurements
5. Jerk – sudden movement acceleration
6. Leading t or f is based on time or frequency measurements
7. Mag – magnitude of movement
8. Mean – average value calculated for each subject for each activity
9. SD – standard deviation value calculated for each subject for each activity

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.

* tBodyAcc-XYZ
* tGravityAcc-XYZ
* tBodyAccJerk-XYZ
* tBodyGyro-XYZ
* tBodyGyroJerk-XYZ
* tBodyAccMag
* tGravityAccMag
* tBodyAccJerkMag
* tBodyGyroMag
* tBodyGyroJerkMag
* fBodyAcc-XYZ
* fBodyAccJerk-XYZ
* fBodyGyro-XYZ
* fBodyAccMag
* fBodyAccJerkMag
* fBodyGyroMag
* fBodyGyroJerkMag

**The set of variables that were estimated from these signals are:**

mean(): Mean value

std(): Standard deviation

**Data Set Information**

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.

**Attribute Information**

For each record in the dataset 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.

- Its activity label.

- An identifier of the subject who carried out the experiment.

**Download Data for Analysis**

## Set working director for the files

setwd("C:/Users/tljon/datasciencecoursera")

## Create file location

if(!file.exists("./data")) {dir.create("./data")}

## Connect to file location on the internet

download.file("https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip",

destfile ="./data/Dataset.zip", method = "curl")

## Unzip the downloaded Dataset.zip to the ./data directory

unzip(zipfile = "./data/Dataset.zip", exdir = "./data")

## Now checking to ensure Dataset.zip has been unzipped and files in folder

list.files("./data")

**Set Path for New Folder**

## Set path for new folder

pathdata = file.path("./data", "UCI HAR Dataset")

files = list.files(pathdata, recursive = TRUE)

## View the files in the list

files

**Load Packages to Support Analysis**

## Load packages to support analysis

library(dplyr)

library(data.table)

library(tidyr)

**Files in the “UCI HAR Dataset”**

1. Test Files
   1. X\_test.txt
   2. Y\_test.txt
   3. Subject\_test.txt
2. Train Files
   1. X\_train.txt
   2. Y\_train.txt
   3. Subject\_train.txt
3. Features.txt – names of column variables in the data table
4. Activity\_lables.txt – links the class labels with their activity name
5. **Merge the training and the test set to create one data set**

## Read the test data set

xtest = read.table(file.path(pathdata, "test", "X\_test.txt"), header = FALSE)

ytest = read.table(file.path(pathdata, "test", "y\_test.txt"), header = FALSE)

subject\_test = read.table(file.path(pathdata, "test", "subject\_test.txt"), header = FALSE)

## Reading the training data set

xtrain = read.table(file.path(pathdata, "train", "X\_train.txt"), header = FALSE)

ytrain = read.table(file.path(pathdata, "train", "y\_train.txt"), header = FALSE)

subject\_train = read.table(file.path(pathdata, "train", "subject\_train.txt"), header = FALSE)

## Read the features data set

features = read.table(file.path(pathdata, "features.txt"), header = FALSE)

## Read the activity labels data set

activityLabels = read.table(file.path(pathdata, "activity\_labels.txt"), header = FALSE)

## Uses descriptive activity names to name the activities in the data set

colnames(xtest) = features[,2]

colnames(ytest) = "activityId"

colnames(subject\_test) = "subjectId"

colnames(xtrain) = features[,2]

colnames(ytrain) = "activityId"

colnames(subject\_train) = "subjectId"

colnames(activityLabels) <- c('activityId', 'activityType')

## Data sets have been created and columns have been named.

## Time to merge the test and train data sets

mergeTest = cbind(ytest, subject\_test, xtest)

mergeTrain = cbind(ytrain, subject\_train, xtrain)

## Create the data table that merges both data sets

MergedSet = rbind(mergeTest, mergeTest)

## This step completes the merger of the data sets

1. **Extract only the measurements on the mean and standard deviation for each measurement**

## Read the values from the new MergedSet and get a subset of all means and standard deviations

colNames = colnames(MergedSet)

MeanAndSD = (grepl("activityId", colNames) | grepl("subjectId", colNames) |

grepl("mean..", colNames) | grepl("std..", colNames))

## Pull the required data set

setMeanAndSD <- MergedSet[, MeanAndSD == TRUE]

1. **Use descriptive activity names to name the activities in the date set**

## Name the activities in the data set using descriptive names

setActivityNames = merge(setMeanAndSD, activityLabels, by = 'activityId', all.x = TRUE)

## The outcomes and solutions are MergedSet and setMeanAndSD

1. **Appropriately labels the data set with descriptive variable names.**

## Create a new tidy set

newTidySet <- aggregate(. ~subjectId + activityId, setActivityNames, mean)

newTidySet <- newTidySet[order(newTidySet$subjectId, newTidySet$activityId), ]

**5. From the data set in step 4, creates a second, independent tidy data set with the average of each variable for each activity and each subject.**

## Write the output to a new saved text file

write.table(newTidySet, "newTidySet.txt", row.name = FALSE)