CodeBook for Week 3 Project

Getting and Cleaning Data Course Coursera

Date: 21 Feb 2015

Author: Randeep Grewal

Project summary Given a set of raw data files comprising of Samsung Galaxy S measurements, in two data sets (training and test) combine them and process them to a tidy dataset.

Then select the columns comprising mean or standard deviation data and average them by subject and group.

Further information on the data is at [http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones]

Raw Data Files The raw data is downloaded from the following url [http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones]

After unzipping it created a directory 'UCI HAR Dataset'

Task details The following is from the project task website

> 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.

Code

My code is broken into steps as above plus step 0. However my steps are in a slightly different order from the task as I felt it was more logical (but for consistency I kept the original task step numbers).

Step 0 - Downloading raw data, unzipping and loading into R This is fairly self-explanatory from the code. I use the following variables:

activity_labels -this provides the label variables we will need later

features -this is the vector which contains the column labels for the X datasets

Steps 1,3,4 - Merge training and test sets, use descriptive activity names and label with descriptive variable names The two subdirectories (test and training) each contain 3 critical files - subject, X and Y These are loaded to *subject_test*, *x_test*, *y_test* or *subject_training*, *x_training* and *y_training* accordingly.

It is important to recognise that the final dataframe we want comprises a cbind of subject, y, x after the test and training datasets have gone through rbind. However the values in y need to be replaced with the values in *activity_labels*. (By *x*, *y* and *subject* I mean *x_test* and *x_training* etc)

The *features* vector contains the labels of the *x* datasets.

For step 4 I chose to use CamelCases. Furthermore I convert the leading 'f' to Freq, and the leading 'T' to Time. For further tidiness and to ensure that the column names are acceptable syntax in R we convert '-std()' to 'Std' and '-mean()' to 'Mean'. The exact code is here: (experimenting re inserting code!)

```
col_names <- names(data)
col_names <- gsub("^t","Time", col_names)
col_names <- gsub("^f","Freq", col_names)
col_names <- gsub("-std\\(\\)-|-std\\(\\)","Std",col_names)
col_names <- gsub("-mean\\(\\)-|-mean\\(\\)","Mean", col_names)

colnames(data) <- col_names
```

Step 2 - Extract only the measurements on the mean and the standard deviation for each measurement I chose to do this step relatively late in my code as it helped me understand what was going on by labelling the dataset first.

There is some discussion in the forums about whether this means we extract every occurrance of mean and standard deviation or only those where the label is mean() or std(). Based on the discussion in the forum I chose to chose only the latter option. See [https://class.coursera.org/getdata-011/forum/thread?thread_id=19]

My key decision is reflected in these line:

```
cols_to_select <- grep("[Mm]ean\\(\\)|-std\\(\\)", colnames(data))
data <- data[,c(1:2,cols_to_select)]
```

By changing the first of these two lines we could change the number of columns the code returns.

By using the above options I get 68 columns - of which 66 are the data from the X dataframes and the other two are subject (= the previous subject dataset) and activity (= the previous Y dataset). Others, by using more permissive searches for 'mean' and 'std' have obtained upto 79 columns (which I did before I looked at the data columns and decided to exclude columns such as 'angle(tBodyGyroJerkMean,gravityMean)')

Step 5 - Independent tidy data set with the average of each varible for each activity and each subject My interpretation of this is that this requires applying the mean function on a dataset that is grouped by both activity and subject. Hence given that there are 30 subjects, each of whom does 6 activities we will get 180 values which indeed the output dataset delivers.

The key lines of code are:

```
newdata <- group_by(data, Subject,Activity)
data_output <- summarise_each(newdata,funs(mean))
```

The dataset *data_output* is written to a file using write.table. I believe it is a tidy dataset as it has 180 rows (ie 30 subjects x 6 activities). There are 68 columns with each column representing the mean of a measured variable.

Code Book

The list below provides the variables that the code writes to 'tidy_data.txt'.

Summar The first variable *Subject* is the id for the subject. The second varible *Activity* describes the activity ("WALKING","WALKING_UPSTAIR","WALKING_DOWNSTAIRS","SITTING","STANDING","LAYING").

For all the other variables the data is the means of the Time or Freq based data from the accelerometer or gyroscope within the phone.

The following is the complete list of the variables (the variable number is for convenience only and not part of the variable name):

1 Subject

2 Activity

3 TimeBodyAccMeanX

4 TimeBodyAccMeanY

5 TimeBodyAccMeanZ

6 TimeBodyAccStdX

7 TimeBodyAccStdY

8 TimeBodyAccStdZ

9 TimeGravityAccMeanX

10 TimeGravityAccMeanY

11 TimeGravityAccMeanZ

12 TimeGravityAccStdX

13 TimeGravityAccStdY

14 TimeGravityAccStdZ

15 TimeBodyAccJerkMeanX

16 TimeBodyAccJerkMeanY

17 TimeBodyAccJerkMeanZ

18 TimeBodyAccJerkStdX

19 TimeBodyAccJerkStdY

20 TimeBodyAccJerkStdZ

21 TimeBodyGyroMeanX

22 TimeBodyGyroMeanY

23 TimeBodyGyroMeanZ

24 TimeBodyGyroStdX

25 TimeBodyGyroStdY

26 TimeBodyGyroStdZ

27 TimeBodyGyroJerkMeanX

28 TimeBodyGyroJerkMeanY

29 TimeBodyGyroJerkMeanZ

30 TimeBodyGyroJerkStdX

31 TimeBodyGyroJerkStdY

32 TimeBodyGyroJerkStdZ

33 TimeBodyAccMagMean

34 TimeBodyAccMagStd

35 TimeGravityAccMagMean

36 TimeGravityAccMagStd

37 TimeBodyAccJerkMagMean

38 TimeBodyAccJerkMagStd

39 TimeBodyGyroMagMean

40 TimeBodyGyroMagStd

41 TimeBodyGyroJerkMagMean

42 TimeBodyGyroJerkMagStd

43 FreqBodyAccMeanX

44 FreqBodyAccMeanY

45 FreqBodyAccMeanZ

46 FreqBodyAccStdX

47 FreqBodyAccStdY

48 FreqBodyAccStdZ

49 FreqBodyAccJerkMeanX

50 FreqBodyAccJerkMeanY

51 FreqBodyAccJerkMeanZ

52 FreqBodyAccJerkStdX

53 FreqBodyAccJerkStdY

54 FreqBodyAccJerkStdZ

55 FreqBodyGyroMeanX

56 FreqBodyGyroMeanY

57 FreqBodyGyroMeanZ

58 FreqBodyGyroStdX

59 FreqBodyGyroStdY

60 FreqBodyGyroStdZ

61 FreqBodyAccMagMean

62 FreqBodyAccMagStd

63 FreqBodyBodyAccJerkMagMean

64 FreqBodyBodyAccJerkMagStd

65 FreqBodyBodyGyroMagMean

66 FreqBodyBodyGyroMagStd

67 FreqBodyBodyGyroJerkMagMean

68 FreqBodyBodyGyroJerkMagStd