CODEBOOK FOR: GETTING AND CLEANING DATA – COURSE PROJECT

BY: GOPAL VENUGOPAL

PROJECT OVERVIEW

The goal of the project is to use the dataset already assembled by a group of scientists in Italy and process it using the data collection and cleaning principles learned during the course. The end objective is to create a tidy dataset, where each record is an observation and every column is a variable. The dataset can be downloaded from:
https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip

This project takes the dataset assembled by Jorge L. Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto based at the Universita degli Studi di Genova, in Genoa, Italy.
Contacts: **activityrecognition@smartlab.ws** and **www.smartlab.ws**
The dataset was assembled from 30 volunteers performing 6 different activities over a period of time, while wearing a smartphone – a Samsung Galaxy S2. The activities were: Walking, Walking Upstairs, Walking Downstairs, Sitting, Standing and Laying. Using the smartphone's embedded accelerometer and gyroscope the scientists collected triaxial linear acceleration and angular velocity readings for each subject's activities. These were recorded at a constant rate of 50 Hz. The raw data from the instrument were pre-processed using noise filters and then sampled suitably in a fixed width sliding window of 2.56 seconds and overlap, generating 128 readings / window. More detailed information on the dataset and collection are provided in the Readme.txt included with the dataset.

For each record the following 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.

The dataset includes the following files:

- 'README.txt'
- 'features_info.txt': Shows information about the variables used on the feature vector.
- 'features.txt': List of all features.
- 'activity_labels.txt': Links the class labels with their activity name.
- 'train/X_train.txt': Training set.
- 'train/y_train.txt': Training labels – defines the activity code corresponding to the activity labels
- 'test/X_test.txt': Test set.
- 'test/y_test.txt': Test labels – defines the activity code corresponding to the activity labels

- 'train/subject_train.txt' and ‘test/subject_test.txt’ – defines the subject ids that correspond to the x_train.txt and x_test.txt observations.

DATA ANALYSIS

DATA REVIEW

First we had to review the dataset and determine how they all fit together. The data was divided into three main groups.

- Training dataset
- Test dataset
- Common files and definitions

Of the 30 volunteers, data from 21 of them was used to create the training dataset and the remaining 9 provided the data for the test dataset.

- The x_train.txt and x_test.txt files contained for each observation (row) a vector of 561 measurements (described in the features.txt file).

- The y_train.txt and y_test.txt files provided the “Activity” code (number 1 – 6) that could be associated with each record in x_train.txt/x_test.txt files respectively. The activity code could be translated to a meaningful activity using the “activity_labels.txt” file provided among the common files.

- Finally, there are train_subject.txt / test_subject.txt files which gives the volunteer’s ID (1 – 30) corresponding to each observation in the x_train/test.txt files.

- Thus the number of rows in all of the associated files would be the same: x_train.txt, y_train.txt, subject_train.txt would have the same number of rows and the same is true for the test sets as well.

DATA SCOPING

The x_train.txt and x_test.txt have the recordings for 561 variables for each observation. These variables are fully described in the features.txt file provided by the scientists that created this dataset. Of these we are only interested in the observations pertaining to the mean and std-dev, for the observations recorded. The remaining observations provide more granular description of the variations in the dataset such as median absolute deviation, max/min values, quartile ranges, correlation, kurtosis, skewness, etc. But for the purpose of this project it was decided to only focus on following fields that would be in the final dataset:

A total of 66 variables were thus identified from the initial set of 561 variables.

1 tBodyAcc-mean()-X	125 tBodyGyro-std()-Y	270 fBodyAcc-std()-Y
2 tBodyAcc-mean()-Y	126 tBodyGyro-std()-Z	271 fBodyAcc-std()-Z
3 tBodyAcc-mean()-Z	161 tBodyGyroJerk-mean()-X	345 fBodyAccJerk-mean()-X
4 tBodyAcc-std()-X	162 tBodyGyroJerk-mean()-Y	346 fBodyAccJerk-mean()-Y
5 tBodyAcc-std()-Y	163 tBodyGyroJerk-mean()-Z	347 fBodyAccJerk-mean()-Z
6 tBodyAcc-std()-Z	164 tBodyGyroJerk-std()-X	348 fBodyAccJerk-std()-X
41 tGravityAcc-mean()-X	165 tBodyGyroJerk-std()-Y	349 fBodyAccJerk-std()-Y
42 tGravityAcc-mean()-Y	166 tBodyGyroJerk-std()-Z	350 fBodyAccJerk-std()-Z
43 tGravityAcc-mean()-Z	201 tBodyAccMag-mean()	424 fBodyGyro-mean()-X
44 tGravityAcc-std()-X	202 tBodyAccMag-std()	425 fBodyGyro-mean()-Y
45 tGravityAcc-std()-Y	214 tGravityAccMag-mean()	426 fBodyGyro-mean()-Z
46 tGravityAcc-std()-Z	215 tGravityAccMag-std()	427 fBodyGyro-std()-X
81 tBodyAccJerk-mean()-X	227 tBodyAccJerkMag-mean()	428 fBodyGyro-std()-Y
82 tBodyAccJerk-mean()-Y	228 tBodyAccJerkMag-std()	429 fBodyGyro-std()-Z
83 tBodyAccJerk-mean()-Z	240 tBodyGyroMag-mean()	503 fBodyAccMag-mean()
84 tBodyAccJerk-std()-X	241 tBodyGyroMag-std()	504 fBodyAccMag-std()
85 tBodyAccJerk-std()-Y	253 tBodyGyroJerkMag-mean()	516 fBodyBodyAccJerkMag-mean()
86 tBodyAccJerk-std()-Z	254 tBodyGyroJerkMag-std()	517 fBodyBodyAccJerkMag-std()
121 tBodyGyro-mean()-X	266 fBodyAcc-mean()-X	529 fBodyBodyGyroMag-mean()
122 tBodyGyro-mean()-Y	267 fBodyAcc-mean()-Y	530 fBodyBodyGyroMag-std()
123 tBodyGyro-mean()-Z	268 fBodyAcc-mean()-Z	542 fBodyBodyGyroJerkMag-mean()
124 tBodyGyro-std()-X	269 fBodyAcc-std()-X	543 fBodyBodyGyroJerkMag-std()

DATA CLEANUP APPROACH

Create a flat data table that provides a consolidated view of the training and test datasets.

1. The approach described below has been coded in “R”. The R script is in the file runanalysis.R
2. First create a vector of the subject ids, one each for the training data and the test datasets, from the subject_test/train.txt files.
3. Next create another vector of the activity labels for each observation in the training and test datasets, by doing a look up of the activity code in y_test/train.txt files against the activity_labels.txt. Thus we will get a vector of the descriptive activities done by each volunteer, for every record of the observations in the data file.
4. Now the vectors created in 1 and 2 above are combined with the 561 variable measurements data to get two flat tables: one for the training dataset and another for the test dataset. The tables would look as below:

SubjectID	Activity	561 measurements for each activity record								
		Var1	Var2	.	.	.	.	.	.	Var561
1	Walking									
1	Sitting									
1	.									
1	.									

5. The data for the testing and training sets are then merged to get a table with 563 columns. Now the column headings are created from the features.txt file for the 561 variables. To this we add the "SubjectID" and "Activity" column heading for the first two columns. Thus we will now have a single table 563 columns. This is created as a data frame called **allData**.
6. The table **allData** contains all the measurements of all the 561 variables being tracked for an observation. But the scope of our analysis is limited to only those variables that are the mean and std-dev of the measurements. Thus we next create a subset of the allData table using a "grepl" command on the column names to only bring in those columns that contain in its name the following substrings: "mean()", "std-dev()","SubjectID","Activity". Thus we efficiently create a new data frame called tidyData consisting of 10299 rows and 68 columns. This tidyData is ordered by SubjectID and Activity.
7. The final dataset requires us to get the mean for each of these 66 variables, but for a given subject_id and activity. Thus a single subject, could be involved in the same activity over multiple records. Thus using the **dplyr** package and the commands group_by, combine the rows by SubjectID and Activity to get 6 records for each SubjectID. For each of these calculate the mean of the measurements that contribute to this grouped by record using the summarise_each() function and passing the "mean" in the **funs** argument.
8. The final tidy data is stored in a data frame finaltidyData, consisting of 180 records (6 for each of the 30 SubjectIDs that participated in the experiment) and 68 columns, of which 66 gives the mean() and std-dev() for the measurements that were scoped into our analysis.

DATA TABLE DESCRIPTIONS

Here are the data table sizes through the different processes:

Datatable Name	Description	Rows	Columns
xtrain	7352 observations, each having 561 variables	7352	561
ytrain	Activity codes for each of the xtrain observations	7352	1
subject_train	subjectIDs for the 7352 observations	7352	1
combined_train	Adds the subject and activity columns to the measurements data	7352	563
xtest	2947 observations, each having 561 variables	2947	561
ytest	Activity codes for each of the xtrain observations	2947	1
subject_test	subjectIDs for the 2947 observations	2947	1
combined_test	Adds the subject and activity columns to the measurements data	2947	563
Activity_Labels	Reference Data that maps the activity codes to an activity	6	2
features	Gives the column names for the 561 variable measurements	1	561
allData	Combines the test and train combined datasets	10299	563
tidyData	Subset of allData consisting of the SubjectID, Activity and 66 variables pertaining to the mean and std-dev measurements	10299	68
finaltidyData	Groups tidyData by subjectID and Activity, calculates the mean for each of the 66 variables	180	68

The finaltidyData had the following 66 variables with their mean() and std-dev() averaged over the grouped records:

1 tBodyAcc-mean()-X	125 tBodyGyro-std()-Y	270 fBodyAcc-std()-Y
2 tBodyAcc-mean()-Y	126 tBodyGyro-std()-Z	271 fBodyAcc-std()-Z
3 tBodyAcc-mean()-Z	161 tBodyGyroJerk-mean()-X	345 fBodyAccJerk-mean()-X
4 tBodyAcc-std()-X	162 tBodyGyroJerk-mean()-Y	346 fBodyAccJerk-mean()-Y
5 tBodyAcc-std()-Y	163 tBodyGyroJerk-mean()-Z	347 fBodyAccJerk-mean()-Z
6 tBodyAcc-std()-Z	164 tBodyGyroJerk-std()-X	348 fBodyAccJerk-std()-X
41 tGravityAcc-mean()-X	165 tBodyGyroJerk-std()-Y	349 fBodyAccJerk-std()-Y
42 tGravityAcc-mean()-Y	166 tBodyGyroJerk-std()-Z	350 fBodyAccJerk-std()-Z
43 tGravityAcc-mean()-Z	201 tBodyAccMag-mean()	424 fBodyGyro-mean()-X
44 tGravityAcc-std()-X	202 tBodyAccMag-std()	425 fBodyGyro-mean()-Y
45 tGravityAcc-std()-Y	214 tGravityAccMag-mean()	426 fBodyGyro-mean()-Z
46 tGravityAcc-std()-Z	215 tGravityAccMag-std()	427 fBodyGyro-std()-X
81 tBodyAccJerk-mean()-X	227 tBodyAccJerkMag-mean()	428 fBodyGyro-std()-Y
82 tBodyAccJerk-mean()-Y	228 tBodyAccJerkMag-std()	429 fBodyGyro-std()-Z
83 tBodyAccJerk-mean()-Z	240 tBodyGyroMag-mean()	503 fBodyAccMag-mean()
84 tBodyAccJerk-std()-X	241 tBodyGyroMag-std()	504 fBodyAccMag-std()
85 tBodyAccJerk-std()-Y	253 tBodyGyroJerkMag-mean()	516 fBodyBodyAccJerkMag-mean()
86 tBodyAccJerk-std()-Z	254 tBodyGyroJerkMag-std()	517 fBodyBodyAccJerkMag-std()
121 tBodyGyro-mean()-X	266 fBodyAcc-mean()-X	529 fBodyBodyGyroMag-mean()
122 tBodyGyro-mean()-Y	267 fBodyAcc-mean()-Y	530 fBodyBodyGyroMag-std()
123 tBodyGyro-mean()-Z	268 fBodyAcc-mean()-Z	542 fBodyBodyGyroJerkMag-mean()
124 tBodyGyro-std()-X	269 fBodyAcc-std()-X	543 fBodyBodyGyroJerkMag-std()

Here is a sample output from the code:

	SubjectID (int)	Activity (fctr)	tBodyAcc-mean()-X (dbl)	tBodyAcc-mean()-Y (dbl)
1	1	WALKING	0.2773308	-0.017383819
2	1	WALKING_UPSTAIRS	0.2554617	-0.023953149
3	1	WALKING_DOWNSTAIRS	0.2891883	-0.009918505
4	1	SITTING	0.2612376	-0.001308288
5	1	STANDING	0.2789176	-0.016137590
6	1	LAYING	0.2215982	-0.040513953
7	2	WALKING	0.2764266	-0.018594920
8	2	WALKING_UPSTAIRS	0.2471648	-0.021412113
9	2	WALKING_DOWNSTAIRS	0.2776153	-0.022661416
10	2	SITTING	0.2770874	-0.015687994
11	2	STANDING	0.2779115	-0.018420827
12	2	LAYING	0.2813734	-0.018158740