COURSERA’S “GETTING AND CLEANING DATA” CODE BOOK

IINDEX.  
  
  
1INTRODUCTION  
2 PREVIOUS CONSIDERATION.  
3 OVERVIEW.  
4 RECODING AND MERGING THE DATA.

5 ASIGNING NAMES. LABELS  
6 ASIGNING NAMES. VARIABLES.  
7 EXTRACTING THE MEAN AND STANDAR DEVIATION MEASUREMENTS.  
8 CALCULATE THE AVERAGE OF THE MEAN AND THE STANDAR DEVIATION FOR EAC ACTIVITY AND SUBJECT.

9 VARIABLES

1INTRODUCTION  
  
This code book it’s made for the project of the coursera’s course “getting and cleaning data”. <https://class.coursera.org/getdata-013/human_grading>  
  
The data that we will use as a base can be downloaded from here: <https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip>  
  
to work properly, the data must be extracted don your working directory. In simpler words, contain the “UCI HAR Dataset” directory and all it’s content must be on your working directory.

SYSTEM.

OS: WINDOWS 7.   
Processor: Intel core i5 3,2 GHz  
RAM: 4 GB.

R studio Version 0.98.1102  
  
Important note: if you are using another OS, it’s possible that you need to change the path format in the scripts in order to work.

2 PREVIOUS CONSIDERATION.  
  
This exercise could be done in a lot of ways. Although we can just merge the data on one simple data frame, we are going to do a little more complicate.  
  
We are going to maintain the difference between the “train” set (view the next point) and the “test” set. To achieve this, we are going to recode some data to avoid the overlapping of label values.  
  
According to the readme, it’s possible to avoid this, and just merging the data on a simpler and easier way but on my personal experience it’s always a good idea, at least on a first approach, maintain the original groups. This let you to review the data and detect possible biases, etc.  
  
So, as we see later, individuals from the test group will be renumbered to 31 to 54 (instead of 1 to 24), and the label activity of these individuals will include the word “TEST”.

3 OVERVIEW.

The main idea of the project it’s to take six data files, merge them in one single data set, and later rename the rows, the columns and work with them.  
  
On first place we have two set of data: the “train” set (with all the files inside the train directory) and the “test” set (with all the files inside the test directory).Inside of these two sets there is 3 different files.

The “subject” file (subject\_test/subject\_train. txt file) indicates who made the test using a number. In the train dataset, subject goes from 1 to 30. In the test dataset, goes from 1 to 24  
  
The “activity label” file (y\_test/y\_train .txt file), that indicates that labels what activity it’s measuring, using numbers from 1 to 6.  
  
The “value” file. (X\_test/X\_train .txt file). Note that the X on the file name it’s on capital letters. It has the results of the 561 variables that has been measured.   
we are going to merge the same kind of files in one Colum (y\_test.txt with y\_train.txt, X\_test with X\_train, and subject\_test with subject\_train) and later, we will create a data frame merging these 3 lectures using the cbind command.

4 RECODING AND MERGING THE DATA.  
  
To avoid confusions and overlapping with the two set of data, the first thing that we are going to do it’s recoding the y\_test.txt values, and the subject\_test.txt values  
  
On this way we create the yrecoded and subjectrecoded variables:  
  
yrecoded <- read.csv (".\\UCI HAR Dataset\\test\\y\_test.txt", header = FALSE) +6

subjectrecoded <- read.csv (".\\UCI HAR Dataset\\test\\subject\_test.txt", header = FALSE) +30  
  
  
With this recoding, the different activity label from the test group it’s now a number between 7 and 12, and the subjects of the test group are numbered from 31 to 54. This will allow us to differentiate it from the activity labels of the train group (from 1 to 6) and from the subjects of the train group (from 1 to 30).

After this, we will merge the files using the rbind command creating 3 variables. Train dataset values are up, and test dataset values down.  
  
traininglabels <- a column with all the values of the activity labels. From 1 to 12

subject <- a column with a number indicating the different subjects. From 1 to 54.

variables <- 561 columns with the measures.  
And finally we just merge these into a data frame, using cbind.  
  
totaldata <- as.data.frame(cbind(subject,traininglabels,variables))

5 ASIGNING NAMES. LABELS  
  
The activity\_labels.txt of our data has the following information:   
  
1 WALKING

2 WALKING\_UPSTAIRS

3 WALKING\_DOWNSTAIRS

4 SITTING

5 STANDING

6 LAYING  
  
we must remember that we have recoded the activity labels value. From 1 to 6 the activity correspond to the train dataset, and from 7 to 12 to the test data set.  
  
we will create the vector:   
  
activitylabels <- c("WALKING",”WALKING\_UPSTAIRS","WALKING\_DOWNSTAIRS","SITTING","STANDING","LAYING","TEST WALKING", "TEST WALKING\_UPSTAIRS","TEST WALKING\_DOWNSTAIRS","TEST SITTING","TEST STANDING","TEST LAYING")  
  
It’s important to note that we maintain the order. From 1 to 6 there are just the original names, and from 7 to 12 are the same original names but we added the word ”TEST” to differentiate the data that coming from one dataset or another. So “walking” will correspond to the element number 1 of the vector, “sitting” with the vector number 4, and “test walking” with the element number 7 of the vector.  
  
After this we just change the the value of the activity label with is equivalent position on the vector.  
  
For example, if in a row we have a value of activity label “4” we just change them with the element number 4 of the vector, “sitting”

6 ASIGNING NAMES. VARIABLES.  
  
To do this, we will read the file “feature.txt”.   
  
variablelist <- read.table(".\\UCI HAR Dataset\\features.txt", header = FALSE)  
  
this file it’s a table, and on the second column it’s the names of the variables. We will select this column and coerce it into a vector. We will add this vector to another vector with the values “subject” and “activity”, to put a name to all the columns.  
  
variablevector <- as.vector(variablelist[,2])

vectornames <- c("subject","activity",(variablevector))

names (totaldata) <- (vectornames)  
  
We must admit that the names of the variables sometimes are not very self-descriptive , but these are the names provided to us. I must admit that I don’t fully understand some of the names, because English it’s not my home language. For the purposes of this project we take these names as good enough for an example.

7 EXTRACTING THE MEAN AND STANDAR DEVIATION MEASUREMENTS.  
  
This part it’s a little ambiguous on the instructions.  
  
We have extracted the columns that has mean() and sd() at the end of the name, using the grep command. And of course we mantain the colums of subject and activity.  
  
tidydata <- totaldata[,c(1,2,grep("mean()",names(totaldata),fixed = TRUE),grep("std()",names(totaldata),fixed = TRUE))]  
  
  
  
  
  
8 CALCULATE THE AVERAGE OF THE MEAN AND THE STANDAR DEVIATION FOR EAC ACTIVITY AND SUBJECT.  
  
This part it’s relatively easy. On first place, we just use the aggregate command, that allow us to subset some parts of the matrix using a criterium (on this case, activitys and subjects), and merge them using a function (on this case, the mean).  
  
tidydirty = aggregate(extractingdata, by=list(activity = extractingdata$activity, subject=extractingdata$subject), mean)  
  
Later we just remove the new subject and activity column that we have obtained (That are full of NAs, because a mean on them has no sense)

tidy = cbind (tidydirty[,1:2],tidydirty[,5:70])

9 VARIABLES

**Activity:** Describes the kind of activity that was done during the measurement. We differentiate the test data group with the word TEST on the first place.  
  
**Subject:** indicates to what subject belongs the measurement using numbers. Subjects from 1 to 30 belong to the train data group, and subjects from 31 to 54 belong to the test data group.  
  
**Means and standar deviation variables**: indicate the mean of the different measures for every activity and subject of these variables. We just don’t understand enough about the original experiment to indicate the units.  
  
If you want to see all the measures from where the means were extracted, just type “extractingdata” after running the script.   
  
If you want to see a full list of the data including the variables not selected, just type “totaldata” after running the script.  
  
IMPORTANT NOTE: As we say before the names of the variables are not very self-descriptive, but are the names provided to us. We just don’t understand enough about the original experiment to change the names. If you want to achieve more information about this, you can go to here: <http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones>