Codebook

Daniel Escasa

2021 January 27

`{r setup, include=FALSE} knitr::opts_chunk$set(echo = TRUE)` # Introduction {-} This book is an adjunct to the run_analysis.R script. It describes the data files, variables, and processes necessary to obtain the following:

1. a tidy dataset
2. training and test sets merged into one
3. the measurements on the mean and standard deviation for each measurement
4. descriptive activity names to name the activities in the data set
5. appropriate labels for the data set with descriptive variable names
6. a second, independent tidy data set with the average of each variable for each activity and each subject.

The data

The dataset is a zip file downloaded from the University of California at Irvine. It extracts to a directory named `UCI HAR Dataset`. Below is its directory structure:

```
-rw-r--r-- 1 daniel daniel     80 Oct 10  2012 activity_labels.txt
-rw-r--r-- 1 daniel daniel   2809 Oct 15  2012 features_info.txt
-rw-r--r-- 1 daniel daniel  15785 Oct 11  2012 features.txt
-rw-r--r-- 1 daniel daniel 635204 Jan 25 10:37 README.html
-rw-r--r-- 1 daniel daniel   4582 Jan 25 10:37 README.md
-rw-r--r-- 1 daniel daniel   4453 Dec 10  2012 README.txt
drwx------ 3 daniel daniel   4096 Nov 29  2012 test
drwx------ 3 daniel daniel   4096 Nov 29  2012 train
```

In `UCI HAR Dataset/test` are the following files:

```
drwx------ 2 daniel daniel     4096 Nov 29  2012 'Inertial Signals'
-rw-r--r-- 1 daniel daniel     7934 Nov 29  2012  subject_test.txt
-rw-r--r-- 1 daniel daniel 26458166 Nov 29  2012  X_test.txt
-rw-r--r-- 1 daniel daniel     5894 Nov 29  2012  y_test.txt
```

and in the `Inertial Signals` directory:

```
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_acc_x_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_acc_y_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_acc_z_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_gyro_x_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_gyro_y_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 body_gyro_z_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 total_acc_x_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 total_acc_y_test.txt
-rw-r--r-- 1 daniel daniel 6041350 Nov 29  2012 total_acc_z_test.txt
```

The structure of the `UCI HAR Dataset/train` directory is similar with the files named `*train` instead of `*test`.

1. The file `activity_labels.txt`, as the name implies, is the file of labels of the activities, i.e.:

```
$ cat activity_labels.txt
1 WALKING
2 WALKING_UPSTAIRS
3 WALKING_DOWNSTAIRS
4 SITTING
5 STANDING
6 LAYING
```

The file `features_info.txt` describes the entries in `features.txt` and how they were derived. Large parts of it are highly technical and mainly of academic interest.

The file `features.txt`, as the name implies, contains the features of interest. The entries therein will, after some manipulations through `mutate()`, become the column names for the required `tidy_summary.txt`.

- Below are the first 15 entries:

```
   V1                V2
1   1 tBodyAcc-mean()-X
2   2 tBodyAcc-mean()-Y
3   3 tBodyAcc-mean()-Z
4   4  tBodyAcc-std()-X
5   5  tBodyAcc-std()-Y
6   6  tBodyAcc-std()-Z
7   7  tBodyAcc-mad()-X
8   8  tBodyAcc-mad()-Y
9   9  tBodyAcc-mad()-Z
10 10  tBodyAcc-max()-X
11 11  tBodyAcc-max()-Y
12 12  tBodyAcc-max()-Z
13 13  tBodyAcc-min()-X
14 14  tBodyAcc-min()-Y
15 15  tBodyAcc-min()-Z
```

- Further down are features that begin with `f`. As explained in`features_info`, the `t` and `f` prefixes refer to time and frequency domains, respectively.
- Note that the first three contain the string "mean", the next three "std". Those signify that those features will produce the means and standard deviations, respectively, of the measurements.
- Finally, the dataset consists of text files, which means that we have to use `tibble::as_tibble(read.table())` and provide column names to enable us to treat the file as data frame.

Massaging the data files

Transforming the`features` file

Examining the `features` file reveals some work to be done:

1. determining the `features` that represent means and standard deviations.
2. replacing the prefixes `t` and `f` with more descriptive `Time` and `Frequency`, respectively
3. replacing "angle" with "Angle" for consistency in capitalization
4. replacing double occurrences of "Body" — i.e., "BodyBody" — with "Body"
5. separating "Acc", "Gyro", "Jerk", and "Mag" from the rest of the feature with dots — i.e., ".Acc", ".Gyro", ".Jerk", and ".Mag"
6. replacing double occurrences of the dot (".") with a single dot

7. removing a dot at the end of the feature.

This long function chain performs this

```
features <- features %>%
    mutate(Is.Mean          = grepl("mean\\(\\)", features$Feature)) %>%
    mutate(Is.Std           = grepl("std\\(\\)", features$Feature)) %>%
    mutate(Feature.Variable = make.names(features$Feature, unique = TRUE)) %>%
    mutate(Feature.Variable = gsub("^t", "Time.", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("\\.t", ".Time.", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("^f", "Frequency.", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("\\.f", ".Frequency.", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("^angle\\.", "Angle.", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("BodyBody", "Body", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("Acc", ".Acc", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("Gyro", ".Gyro", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("Jerk", ".Jerk", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("Mag", ".Mag", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("\\.\\.", ".", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("\\.\\.", ".", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("\\.$", "", Feature.Variable)) %>%
    mutate(Feature.Variable = gsub("(^|[\\.])([[:alpha:]])", "\\1\\U\\2",
                    Feature.Variable, perl=TRUE))
```

mutate() rocks!

Loading the activities

This is a simple matter of invoking `tibble::as_tibble(read.table())` on `activity_labels.txt`, and assigning column names.

Loading the training dataset

This consists of four steps:

1. setting `train` column names to the rows of `features` dataset
2. adding subject data and activity data to the training dataset
3. renaming the V1 columns to `Subject.Id` in `subject_training.txt` and to `Activity.Id` in y_training.txt
4. combining the two txt files above with `cbind()`

Loading the test dataset

Activity here is the same as in the previous section, applied to the test dataset.

Merge the training and test datasets

This consists of the following steps:

1. adding descriptive activity names from activities
2. selecting the mean and std deviation features only.
3. grouping by subject and activity.
4. merging the training and test datasets using rbind() into a `merged` dataset

Creating the tidy summary

The `summary()` function, applied to the `merged` training and test databases from the previous section, will create the required `tidy_summary` file.

Writing the tidy summary to a file

A simple `write.table()` applied to `tidy_summary` wraps up the project.