Human Activity Recognition Using Smartphones Dataset

Clement Liu

December 27, 2015

Overview

The following script tidies the dataset provided by the experiments carried out by **Jorge L. Reyes-Ortiz**, **Davide Anguita**, **Alessandro Ghio**, and **Luca Oneto** of **Smartlab - Non Linear Complex Systems Laboratory** in **Genoa, Italy**. The original files contained the following datasets:

Training	Test
Subject	Subject
Set	Set
Labels	Labels

These datasets would be merged and tidied to create the output: tidydata.txt.

The researchers also included additional files that were used to compile the tidy dataset:

File	Description
README	Overview of the experiment and files
features	List of all the features
features_info	Explains the variables in features
activity_labels	Links class labels with activity name

Text from Researchers' README

The experiments were carried out by 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. See 'features_info.txt' for more details.

Script and Descriptions

The script starts with downloading the data (in zip) from the website. It will create a "data" folder in the directory (if it does not already exist), then unzip the files into a subfolder named "project_data".

```
if (!file.exists("./data")) {
    dir.create("./data")
    fileUrl <- "https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip"
    download.file(fileUrl, destfile = "./data/HAR_Dataset.zip")
    unzip("./data/HAR_Dataset.zip", exdir = "project_data")
}
```

The script will then read all of the relevant datasets and files into appropriate variables.

```
subjecttest <- read.table("./project_data/UCI HAR Dataset/test/subject_test.txt")
xtest <- read.table("./project_data/UCI HAR Dataset/test/X_test.txt")
ytest <- read.table("./project_data/UCI HAR Dataset/test/y_test.txt")
subjecttrain <- read.table("./project_data/UCI HAR Dataset/train/subject_train.txt")
xtrain <- read.table("./project_data/UCI HAR Dataset/train/X_train.txt")
ytrain <- read.table("./project_data/UCI HAR Dataset/train/y_train.txt")
activitylabels <- read.table("./project_data/UCI HAR Dataset/activity_labels.txt")
features <- read.table("./project_data/UCI HAR Dataset/features.txt")
```

The script continues by merging the ancillary data into the main datasets. The dplyr package is necessary to enable the join.

```
library(dplyr)
colnames(xtest) <- features$V2
colnames(xtrain) <- features$V2
xtest$subject <- subjecttest$V1
xtrain$subject <- subjecttrain$V1
ytrain <- left_join(ytrain, activitylabels, by = "V1")
colnames(ytrain) <- c("class_label", "activity_name")
ytest <- left_join(ytest, activitylabels, by = "V1")
colnames(ytest) <- c("class_label", "activity_name")
```

The master dataset is created by first binding the set and labels for both the test and training datasets. The test and training datasets are then bound to create the complete dataset.

```
xtest <- cbind(xtest, ytest)
xtrain <- cbind(xtrain, ytrain)
project_data <- rbind(xtest, xtrain)
```

The script then extracts the dataset for only variables related to means and standard deviations. It will also rename the variables to be more reader-friendly.

```
meanstd <- project_data[, c(grepl("mean\\(\\)|std\\(\\)|subject|activity_name",
    colnames(project_data), ignore.case = T))]
colnames(meanstd)[1:66] <- sub("^t", "Time", colnames(meanstd)[1:66])
colnames(meanstd)[1:66] <- sub("^f", "Freq", colnames(meanstd)[1:66])
colnames(meanstd)[1:66] <- sub("mean", "Mean", colnames(meanstd)[1:66])
colnames(meanstd)[1:66] <- sub("std", "STD", colnames(meanstd)[1:66])
```

```
colnames(meanstd)[1:66] <- sub("\\(", "", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- sub("\\)", "", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- sub("X$", "_X", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- sub("Y$", "_Y", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- sub("Z$", "_Z", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- gsub("\\-", "", colnames(meanstd[1:66]))
colnames(meanstd)[1:66] <- sub("BodyBody", "Body", colnames(meanstd[1:66]))
```

Finally, the script will group the variables by "subject" and "activity_name" and then find the means of each of the other variables. The tidy dataset is created and saved into a text document named "tidydata.txt".

```
by_act_subj <- group_by(meanstd, subject, activity_name)
tidydata <- summarize_each(by_act_subj, funs(mean))
write.table(tidydata, "tidydata.txt", row.names = FALSE)
```