CodeBook

Heinrich Ziegler

10/22/2020

## CodeBook: Peer-graded Assignment: Getting and Cleaning Data Course Project

In order to get and clean/tidy the requested data just run run\_analysis.R

The script follows the 5-step criteria as stated in the assignment description.

However, before processing the data is needs first to be loaded of the internet and then unzipped.

### Downloading the data

downloading…

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.0.3

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

destfile <- 'Coursera\_DS3\_Final.zip'  
   
 if (!file.exists(destfile)){  
 fileURL <- 'https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip'  
   
 download.file(fileURL, destfile, method='curl')  
   
 }

### Unzipping data

The zipped file contains a folder “UCI HAR Dataset” folder. If it already exists then don’t unzip again

if (!file.exists("UCI HAR Dataset")) {   
 unzip(destfile)   
}

### loading and assigning content

Following variables are loaded with the data from the individual files extracted in the previous step

* activities: data.frame; 6 rows, 2 columns; Desc: List of activities represented by Id used in other files as reference
* features: data.frame; 561 rows, 2 columns; list of action processing functions accompanied by and id (n)
* subject\_train: data.frame; subject train data with 7352 rows and 1 column; id of 21 unique train subjects
* x\_train: data.frame; 7352 rows and 561 columns; each column name matches the row values of feature file, thus representing functions
* y-train: data.frame; 7352 rows and 1 column; contains the matching activity codes for the x-train data
* subject\_test: data.frame; subject test data with 2947 rows and 1 column; id of 9 unique test subjects
* x\_test: data.frame; 2947 rows and 561 columns; corresponding test data each column name matches the row values of feature file, thus representing functions
* y-test: data.frame; 2947 rows and 1 column; contains the matching activity codes for the x-test data

activities <- read.table("UCI HAR Dataset/activity\_labels.txt",   
 col.names = c("id", "activity"))  
features <- read.table("UCI HAR Dataset/features.txt",   
 col.names = c("n","functions"))  
subject\_train <- read.table("UCI HAR Dataset/train/subject\_train.txt",   
 col.names = "subject")  
x\_train <- read.table("UCI HAR Dataset/train/X\_train.txt",   
 col.names = features$functions)  
y\_train <- read.table("UCI HAR Dataset/train/y\_train.txt",   
 col.names = "id")  
subject\_test <- read.table("UCI HAR Dataset/test/subject\_test.txt",   
 col.names = "subject")  
x\_test <- read.table("UCI HAR Dataset/test/X\_test.txt",   
 col.names = features$functions)  
y\_test <- read.table("UCI HAR Dataset/test/y\_test.txt",   
 col.names = "id")

### Step 1: Merges the training and the test sets to create one data set

* xAll: 10299 rows, 561 columns after merging X values
* yAll: 10299 rows, 1 column after merging Y values
* subjectAll: 10299 rows, 1 column after merging subject values
* compositeData: 10299 rows, 563 columns after column merging all three data.frames mentioned above

xAll <- rbind(x\_train, x\_test)  
yAll <- rbind(y\_train, y\_test)  
  
subjectAll <- rbind(subject\_train, subject\_test)  
  
compositeData <- cbind(subjectAll, yAll, xAll)

### Step2: Extracts only the measurements on the mean and standard deviation for each measurement

* extractData: 10299 rows and 88 columns, after extracting

extractedData <- select(compositeData, subject, id, contains("mean"), contains("std"))

### Step3: Uses descriptive activity names to name the activities in the data set

* First we change the codes into the actual activities

extractedData$id <- activities[extractedData$id,2]

### Step4: Appropriately labels the data set with descriptive variable names.

* then we clean out abbreviations like acc to Accelerometer and others
* then turning all names into camel case, which makes it more readable to IT personnel

# Appropriately labels the data set with descriptive variable names.  
# removing dots(.) and Transform everything into camel case as used  
# as standards in many programming languages  
  
colnames(extractedData)<-gsub("Acc", "Accelerometer", colnames(extractedData))  
colnames(extractedData)<-gsub("^t", "Time", colnames(extractedData))  
colnames(extractedData)<-gsub("\\.mean", "Mean", colnames(extractedData))  
colnames(extractedData)<-gsub("Freq", "Frequency", colnames(extractedData))  
colnames(extractedData)<-gsub("^f", "Frequency", colnames(extractedData))  
colnames(extractedData)<-gsub("Gyro", "Gyroscope", colnames(extractedData))  
colnames(extractedData)<-gsub("BodyBody", "Body", colnames(extractedData))  
colnames(extractedData)<-gsub("Mag", "Magnitude", colnames(extractedData))  
colnames(extractedData)<-gsub("\\.gravity", "Gravity", colnames(extractedData))  
colnames(extractedData)<-gsub("Acc", "Accelerometer", colnames(extractedData))  
colnames(extractedData)<-gsub("^t", "Time", colnames(extractedData))  
colnames(extractedData)<-gsub("\\.mean", "Mean", colnames(extractedData))  
colnames(extractedData)<-gsub("Freq", "Frequency", colnames(extractedData))  
colnames(extractedData)<-gsub("^f", "Frequency", colnames(extractedData))  
colnames(extractedData)<-gsub("Gyro", "Gyroscope", colnames(extractedData))  
colnames(extractedData)<-gsub("BodyBody", "Body", colnames(extractedData))  
colnames(extractedData)<-gsub("Mag", "Magnitude", colnames(extractedData))  
colnames(extractedData)<-gsub("^anglet", "AngleTime", colnames(extractedData))  
colnames(extractedData)<-gsub("^angle", "Angle", colnames(extractedData))  
colnames(extractedData)<-gsub("std", "StandardDeviation", colnames(extractedData))  
  
# now remove the remaining dots  
colnames(extractedData)<-gsub("\\.", "", colnames(extractedData))  
  
# adapt the first two columns to match  
colnames(extractedData)[1] = "Subject"  
colnames(extractedData)[2] = "Activity"  
# now remove the remaining dots  
colnames(extractedData)<-gsub("\\.", "", colnames(extractedData))

### Step 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.

write.table(extractedData, "TidyExtractedData.txt", row.name=FALSE)  
SummaryData <- extractedData %>% group\_by(Subject, Activity) %>% summarise\_all(list(mean))  
write.table(SummaryData, "SummaryData.txt", row.name=FALSE)

SummaryData contains the result data