Getting and Cleaning Smartphone Accelerometer Data

Matthew Dunne

Assignment

One of the most exciting areas in all of data science right now is wearable computing - see for example this article . Companies like Fitbit, Nike, and Jawbone Up are racing to develop the most advanced algorithms to attract new users. The data linked to from the course website represent data collected from the accelerometers from the Samsung Galaxy S smartphone. A full description is available at the site where the data was obtained:
http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones

Here are the data for the project:
https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip

You should create one R script called run_analysis.R that does the following.

1. Merges the training and the test sets to create one data set.

2. Extracts only the measurements on the mean and standard deviation for each measurement.

3. Uses descriptive activity names to name the activities in the data set.

4. Appropriately labels the data set with descriptive variable names.

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.

```
run_analyis<-function(){
  setwd("C:/Users/mjdun/Desktop/Coursera/Data Science Specialization/Course 3 Getting and Cleaning Data/
  var_names<-read.table("features.txt", stringsAsFactors = FALSE)

  X_test<-read.table("X_test.txt")
  ## takes features and puts them as column names for X_test, can't subset on column names without givin
  names(X_test)<-var_names[ ,2]
  ## subset on desired mean/st. dev. Measurements and name DF test_data
  test_data<-X_test[grep("std|-[Mm]ean", names(X_test), value = TRUE)]

  ##read in subject and activity number columns
  sub_test<-read.table("subject_test.txt")
  act_num1<-read.table("y_test.txt")

  ## puts sub_test and act_num as front columns of test_data
  test_data<-data.frame(sub_test, act_num1, test_data)

  ## read in training data and put labels on it
  X_train<-read.table("X_train.txt")
  ## takes features and puts them as column names for X_train, otherwise rbind won't work
  names(X_train)<-var_names[ ,2]
  ## subset on desired mean/st. dev. Measurements and name DF train_data
  train_data<-X_train[grep("std|-[Mm]ean", names(X_train), value = TRUE)]

  ## puts sub_train and act_num2 as front columns of X_train
  sub_train<-read.table("subject_train.txt")
  act_num2<-read.table("y_train.txt")
  train_data<-data.frame(sub_train, act_num2, train_data)

  ## merge both datasets
  data<-rbind(test_data, train_data)

  actlab<-read.table("activity_labels.txt")
  ## have to load qdapTools package for lookup function
  library(qdapTools)
  ## apply descriptive names from
  data$V1.1<-lookup(data$V1.1, actlab)

  ## renames first two columns subject and activity respectively
  names(data)[1]<-"subject"
  names(data)[2]<-"activity"

  library(dplyr)
  ## group by subject, activity and apply mean function to each subsequent column (summarize would only
  grouped_data<-data %>% group_by(subject, activity) %>% summarize_each(funs(mean))
  write.table(grouped_data, file="Matt's tidy data.txt", row.names = FALSE)
}
```

When you run the function you will create the file “Matt’s tidy data.txt” in the same folder from which you drew the data.

```
run_analyis()
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:qdapTools':
##
##     id

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

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

## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over all variables, use `summarise_all()`
```