Rmd codebook

the codebook has been created in RMD this in order to combine both the code and the results of the code and to make it easy to rerun the whole process of tidying the data.

At the end of the codebook also the code is included for processing the data as required in the assignment. All the code is also in the run_analysis.R program.

I do not pretend to better understand the data then the makers of the data and therefor rely upon and hence have included the ReadMe information of the makers. I only dared to add a small remark on what I make of the explanation they gave.

Human Activity Recognition Using Smartphones Data Set

Abstract: Human Activity Recognition database built from the recordings of 30 subjects performing activities of daily living (ADL) while carrying a waist-mounted smartphone with embedded inertial sensors.

Data Set Characteristics:???? Multivariate, Time-Series Number of Instances: 10299 Area: Computer Attribute Characteristics: N/A Number of Attributes: 561 Date Donated 2012-12-10 Associated Tasks: Classification, Clustering Missing Values? N/A Number of Web Hits: 456069

end of Human Activity Recognition Using Smartphones Data Set downloaded on 2017-03-06, ReadMe content and extra readings were removed for brevity

The original Readme

Human Activity Recognition Using Smartphones Dataset Version 1.0

==========

Jorge L. Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto. Smartlab - Non Linear Complex Systems Laboratory DITEN - Universit?? degli Studi di Genova. Via Opera Pia 11A, I-16145, Genoa, Italy. activityrecognition@smartlab.ws (mailto:activityrecognition@smartlab.ws) www.smartlab.ws

==========

The experiments 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.

For each record it is provided:

- Triaxial acceleration from the accelerometer (total acceleration) and the estimated body acceleration.
- Triaxial Angular velocity from the gyroscope.
- A 561-feature vector with time and frequency domain variables.
- Its activity label.
- An identifier of the subject who carried out the experiment.

The dataset includes the following files:

- ‘README.txt’
- ‘features_info.txt’: Shows information about the variables used on the feature vector.
- ‘features.txt’: List of all features.
- ‘activity_labels.txt’: Links the class labels with their activity name.
- ‘train/X_train.txt’: Training set.
- ‘train/y_train.txt’: Training labels.
- ‘test/X_test.txt’: Test set.
- ‘test/y_test.txt’: Test labels.

The following files are available for the train and test data. Their descriptions are equivalent.

- ‘train/subject_train.txt’: Each row identifies the subject who performed the activity for each window sample. Its range is from 1 to 30.
- ‘train/Inertial Signals/total_acc_x_train.txt’: The acceleration signal from the smartphone accelerometer X axis in standard gravity units ‘g’. Every row shows a 128 element vector. The same description applies for the ‘total_acc_x_train.txt’ and ‘total_acc_z_train.txt’ files for the Y and Z axis.
- ‘train/Inertial Signals/body_acc_x_train.txt’: The body acceleration signal obtained by subtracting the gravity from the total acceleration.
- ‘train/Inertial Signals/body_gyro_x_train.txt’: The angular velocity vector measured by the gyroscope for each window sample. The units are radians/second.

Notes:

- Features are normalized and bounded within [-1,1].
- Each feature vector is a row on the text file.

For more information about this dataset contact: activityrecognition@smartlab.ws (mailto:activityrecognition@smartlab.ws)

License:

Use of this dataset in publications must be acknowledged by referencing the following publication [1]

[1] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra and Jorge L. Reyes-Ortiz. Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine. International Workshop of Ambient Assisted Living (IWAAL 2012). Vitoria-Gasteiz, Spain. Dec 2012

This dataset is distributed AS-IS and no responsibility implied or explicit can be addressed to the authors or their institutions for its use or misuse. Any commercial use is prohibited.

Jorge L. Reyes-Ortiz, Alessandro Ghio, Luca Oneto, Davide Anguita. November 2012.

end of Readme

as per my understanding

The detail data described in the Inertial Signals is not used directly. The values of the measurements are in the preprocessing step turned into vector values cleansed for noise then normalised between -1 and 1 in the aggregated files X_train.txt and X_test.txt. (together containing all the data for the experiment)

The final file will contain the calcualted mean of the means and standard deviations of the Signals from the combined X-train and X_test datasets.

description of the tidy process

Research of the data made clear that there were 561 variables available in the data file and this was an equal number to the number of different calculated columns that were available in de features file. Subsequently the number of Records is equal to the number of items in the subject and activity files, for both the test and training set.

Though tempted to separate the angle directed variables (coded with an f at the front) from the more movement driven data points, I have decided not to do this because the angle is another value of the movement as such. Therewith splitting the table would create in my opinion unnatural separation of a single meaning table.

The steps followed. First the file location is set for the different relevant files. The signals files are left aside since only some data on the agregates is requested. The Header information for the numeric variables was captured from “features.txt” and cleaned(from ()) to then create a list. Subsequently the data parts are gathered and the activity-labels data are added to the rows. Next the numeric labels for person and subject were replaced by a text label. Lastly the rows of the complete file combining test and train were identified in the row names by creating the row name combining the number of the row in the subsequent files and the name of the file in front before binding the two datasets together. For clarity a more descriptive name is given to the data.frame. on checking with the dim() function the expected 10229 records are found.

The code for the largely repetitive steps of adding column names, the person related and the activity related columns are indented for clarity in the code, and because I did feel chalenged to create a function, but for only two process sequences and many parameters needed I decided not to do so.

end of description of the tidy process

the actual programmed manipulations follow hereunder

The loading and unzipping is being done only once here, but when the # is removed it will happen again:

load:

``` {r load}

download.file("https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip (https://d396qusza40orc.cloudfront.net/getdata%2Fprojectfiles%2FUCI%20HAR%20Dataset.zip)","phone.zip")

unzip("phone.zip",exdir=".\phones")

```

First setting up the path for reading

```
library(stringr)
pa<-"D:\\git\\phones\\UCI HAR Dataset\\"
tr<-"train\\"
ts<-"test\\"
patr<-paste(pa,tr,sep="")
pats<-paste(pa,ts,sep="")
```

The next step is adding the variable names to the data and adding the categories for the rows (subjects[persons] and activities) then naming the rows to discern the different sourcefiles(if needed) I removed the brackets in the variable names that prevented me from casting them into a list the train and test files have different length, meaning that te subject (person) and activity lists are different for these files have to be loaded and bound before the files are made into a single file.

```
    # the variables are for all the same
vars<-read.csv(paste(pa,"features.txt",sep=""),sep=" ",header=FALSE)
    # and the activities are all the same
act_lab=read.csv(paste(pa,"activity_labels.txt",sep=""),sep=" ",header=FALSE) # then t
he activity lables
vr<-sub("\\(\\)","",vars[,2])   #removed the brackets in varnames that prevented casti
ng them into a list
vr<-as.list(vr)
# the training dataset
x_tr<-read.table(paste(patr,"x_train.txt",sep=""), colClasses="numeric",header=FALSE)
    # indentation to show that these steps are almost identical to above
    suj<-read.table(paste(patr,"subject_train.txt",sep=""),header=FALSE) #getting the
subjects for training files
    srcd<-paste("train",c(1:length(suj[,1])),sep="")

    act<-read.csv(paste(patr,"y_train.txt",sep=""),header=FALSE) # getting the numeric
s for activities
    act<-merge(act,act_lab) # assigning the lables to the rows as fetched above
    x_tr<-cbind(suj,act[,2],x_tr)
    names(x_tr)<-c("person","activity",vr)   # naming the variables fetched above
    row.names(x_tr)<-srcd   # to discern test and train data
dim(x_tr)
```

```
## [1] 7352  563
```

```
    # the training dataset
x_ts<-read.table(paste(pats,"x_test.txt",sep=""), colClasses="numeric",header=FALSE)
    # indentation to show that these steps are almost identical to above
    suj<-read.table(paste(pats,"subject_test.txt",sep=""),header=FALSE) #getting the s
ubjects for training files
    srcd<-paste("test",c(1:length(suj[,1])),sep="")

    act<-read.csv(paste(pats,"y_test.txt",sep=""),header=FALSE) # getting the numerics
for activities
    act<-merge(act,act_lab) # assigning the lables to the rows as fetched above
    x_ts<-cbind(suj,act[,2],x_ts)
    names(x_ts)<-c("person","activity",vr)   # naming the variables fetched above
    row.names(x_ts)<-srcd  # to discern test and train data
    # get together, to prevent memory issue add x_ts to x_tr
x_tr<-rbind(x_tr,x_ts)
rm(x_ts) #clean up
    # then assign() a new name for better quality reference
assign("phonemotion",x_tr)   # a meaninful name for the combined set
dim(phonemotion)
```

```
## [1] 10299   563
```

```
phonemotion<-phonemotion[,grepl("acti|person|mean|std",names(phonemotion))]
str(phonemotion)
```

```
## 'data.frame':    10299 obs. of  81 variables:
##  $ person                      : int  1 1 1 1 1 1 1 1 1 1 ...
       the person on whom the measurement was done
##  $ activity                    : Factor w/ 6 levels "LAYING","SITTING",..:
       the activity lable given
       then follow the mean and stadard deviation measurements in X,Y,Z directions
##  $ tBodyAcc-mean-X             : num  0.289 0.278 0.28 0.279 0.277 ...
##  $ tBodyAcc-mean-Y             : num  -0.0203 -0.0164 -0.0195 -0.0262 -0.0166 ...
##  $ tBodyAcc-mean-Z             : num  -0.133 -0.124 -0.113 -0.123 -0.115 ...
##  $ tBodyAcc-std-X              : num  -0.995 -0.998 -0.995 -0.996 -0.998 ...
##  $ tBodyAcc-std-Y              : num  -0.983 -0.975 -0.967 -0.983 -0.981 ...
##  $ tBodyAcc-std-Z              : num  -0.914 -0.96 -0.979 -0.991 -0.99 ...
##  $ tGravityAcc-mean-X          : num  0.963 0.967 0.967 0.968 0.968 ...
##  $ tGravityAcc-mean-Y          : num  -0.141 -0.142 -0.142 -0.144 -0.149 ...
##  $ tGravityAcc-mean-Z          : num  0.1154 0.1094 0.1019 0.0999 0.0945 ...
##  $ tGravityAcc-std-X           : num  -0.985 -0.997 -1 -0.997 -0.998 ...
##  $ tGravityAcc-std-Y           : num  -0.982 -0.989 -0.993 -0.981 -0.988 ...
##  $ tGravityAcc-std-Z           : num  -0.878 -0.932 -0.993 -0.978 -0.979 ...
##  $ tBodyAccJerk-mean-X         : num  0.078 0.074 0.0736 0.0773 0.0734 ...
##  $ tBodyAccJerk-mean-Y         : num  0.005 0.00577 0.0031 0.02006 0.01912 ...
##  $ tBodyAccJerk-mean-Z         : num  -0.06783 0.02938 -0.00905 -0.00986 0.01678 .
..
##  $ tBodyAccJerk-std-X          : num  -0.994 -0.996 -0.991 -0.993 -0.996 ...
##  $ tBodyAccJerk-std-Y          : num  -0.988 -0.981 -0.981 -0.988 -0.988 ...
##  $ tBodyAccJerk-std-Z          : num  -0.994 -0.992 -0.99 -0.993 -0.992 ...
##  $ tBodyGyro-mean-X            : num  -0.0061 -0.0161 -0.0317 -0.0434 -0.034 ...
##  $ tBodyGyro-mean-Y            : num  -0.0314 -0.0839 -0.1023 -0.0914 -0.0747 ...
##  $ tBodyGyro-mean-Z            : num  0.1077 0.1006 0.0961 0.0855 0.0774 ...
##  $ tBodyGyro-std-X             : num  -0.985 -0.983 -0.976 -0.991 -0.985 ...
##  $ tBodyGyro-std-Y             : num  -0.977 -0.989 -0.994 -0.992 -0.992 ...
##  $ tBodyGyro-std-Z             : num  -0.992 -0.989 -0.986 -0.988 -0.987 ...
##  $ tBodyGyroJerk-mean-X        : num  -0.0992 -0.1105 -0.1085 -0.0912 -0.0908 ...
##  $ tBodyGyroJerk-mean-Y        : num  -0.0555 -0.0448 -0.0424 -0.0363 -0.0376 ...
##  $ tBodyGyroJerk-mean-Z        : num  -0.062 -0.0592 -0.0558 -0.0605 -0.0583 ...
##  $ tBodyGyroJerk-std-X         : num  -0.992 -0.99 -0.988 -0.991 -0.991 ...
##  $ tBodyGyroJerk-std-Y         : num  -0.993 -0.997 -0.996 -0.997 -0.996 ...
##  $ tBodyGyroJerk-std-Z         : num  -0.992 -0.994 -0.992 -0.993 -0.995 ...
##  $ tBodyAccMag-mean            : num  -0.959 -0.979 -0.984 -0.987 -0.993 ...
##  $ tBodyAccMag-std             : num  -0.951 -0.976 -0.988 -0.986 -0.991 ...
##  $ tGravityAccMag-mean         : num  -0.959 -0.979 -0.984 -0.987 -0.993 ...
##  $ tGravityAccMag-std          : num  -0.951 -0.976 -0.988 -0.986 -0.991 ...
##  $ tBodyAccJerkMag-mean        : num  -0.993 -0.991 -0.989 -0.993 -0.993 ...
##  $ tBodyAccJerkMag-std         : num  -0.994 -0.992 -0.99 -0.993 -0.996 ...
##  $ tBodyGyroMag-mean           : num  -0.969 -0.981 -0.976 -0.982 -0.985 ...
##  $ tBodyGyroMag-std            : num  -0.964 -0.984 -0.986 -0.987 -0.989 ...
##  $ tBodyGyroJerkMag-mean       : num  -0.994 -0.995 -0.993 -0.996 -0.996 ...
##  $ tBodyGyroJerkMag-std        : num  -0.991 -0.996 -0.995 -0.995 -0.995 ...
##  $ fBodyAcc-mean-X             : num  -0.995 -0.997 -0.994 -0.995 -0.997 ...
##  $ fBodyAcc-mean-Y             : num  -0.983 -0.977 -0.973 -0.984 -0.982 ...
##  $ fBodyAcc-mean-Z             : num  -0.939 -0.974 -0.983 -0.991 -0.988 ...
```

```
##  $ fBodyAcc-std-X               : num  -0.995 -0.999 -0.996 -0.996 -0.999 ...
##  $ fBodyAcc-std-Y               : num  -0.983 -0.975 -0.966 -0.983 -0.98 ...
##  $ fBodyAcc-std-Z               : num  -0.906 -0.955 -0.977 -0.99 -0.992 ...
##  $ fBodyAcc-meanFreq-X          : num  0.252 0.271 0.125 0.029 0.181 ...
##  $ fBodyAcc-meanFreq-Y          : num  0.1318 0.0429 -0.0646 0.0803 0.058 ...
##  $ fBodyAcc-meanFreq-Z          : num  -0.0521 -0.0143 0.0827 0.1857 0.5598 ...
##  $ fBodyAccJerk-mean-X          : num  -0.992 -0.995 -0.991 -0.994 -0.996 ...
##  $ fBodyAccJerk-mean-Y          : num  -0.987 -0.981 -0.982 -0.989 -0.989 ...
##  $ fBodyAccJerk-mean-Z          : num  -0.99 -0.99 -0.988 -0.991 -0.991 ...
##  $ fBodyAccJerk-std-X           : num  -0.996 -0.997 -0.991 -0.991 -0.997 ...
##  $ fBodyAccJerk-std-Y           : num  -0.991 -0.982 -0.981 -0.987 -0.989 ...
##  $ fBodyAccJerk-std-Z           : num  -0.997 -0.993 -0.99 -0.994 -0.993 ...
##  $ fBodyAccJerk-meanFreq-X      : num  0.8704 0.6085 0.1154 0.0358 0.2734 ...
##  $ fBodyAccJerk-meanFreq-Y      : num  0.2107 -0.0537 -0.1934 -0.093 0.0791 ...
##  $ fBodyAccJerk-meanFreq-Z      : num  0.2637 0.0631 0.0383 0.1681 0.2924 ...
##  $ fBodyGyro-mean-X             : num  -0.987 -0.977 -0.975 -0.987 -0.982 ...
##  $ fBodyGyro-mean-Y             : num  -0.982 -0.993 -0.994 -0.994 -0.993 ...
##  $ fBodyGyro-mean-Z             : num  -0.99 -0.99 -0.987 -0.987 -0.989 ...
##  $ fBodyGyro-std-X              : num  -0.985 -0.985 -0.977 -0.993 -0.986 ...
##  $ fBodyGyro-std-Y              : num  -0.974 -0.987 -0.993 -0.992 -0.992 ...
##  $ fBodyGyro-std-Z              : num  -0.994 -0.99 -0.987 -0.989 -0.988 ...
##  $ fBodyGyro-meanFreq-X         : num  -0.2575 -0.0482 -0.2167 0.2169 -0.1533 ...
##  $ fBodyGyro-meanFreq-Y         : num  0.0979 -0.4016 -0.0173 -0.1352 -0.0884 ...
##  $ fBodyGyro-meanFreq-Z         : num  0.5472 -0.0682 -0.1107 -0.0497 -0.1622 ...
##  $ fBodyAccMag-mean             : num  -0.952 -0.981 -0.988 -0.988 -0.994 ...
##  $ fBodyAccMag-std              : num  -0.956 -0.976 -0.989 -0.987 -0.99 ...
##  $ fBodyAccMag-meanFreq         : num  -0.0884 -0.0441 0.2579 0.0736 0.3943 ...
##  $ fBodyBodyAccJerkMag-mean     : num  -0.994 -0.99 -0.989 -0.993 -0.996 ...
##  $ fBodyBodyAccJerkMag-std      : num  -0.994 -0.992 -0.991 -0.992 -0.994 ...
##  $ fBodyBodyAccJerkMag-meanFreq : num  0.347 0.532 0.661 0.679 0.559 ...
##  $ fBodyBodyGyroMag-mean        : num  -0.98 -0.988 -0.989 -0.989 -0.991 ...
##  $ fBodyBodyGyroMag-std         : num  -0.961 -0.983 -0.986 -0.988 -0.989 ...
##  $ fBodyBodyGyroMag-meanFreq    : num  -0.129 -0.272 -0.2127 -0.0357 -0.2736 ...
##  $ fBodyBodyGyroJerkMag-mean    : num  -0.992 -0.996 -0.995 -0.995 -0.995 ...
##  $ fBodyBodyGyroJerkMag-std     : num  -0.991 -0.996 -0.995 -0.995 -0.995 ...
##  $ fBodyBodyGyroJerkMag-meanFreq: num  -0.0743 0.1581 0.4145 0.4046 0.0878 ...
```

All has been brought together in one single data.frame.

Now calculate the mean value for all colums containing mean and std values as required

```
phonemotion<-with(phonemotion,phonemotion[order(phonemotion[,1],phonemotion[,2]),])
    # The aggregate function also attempts to aggregate the grouping columns, which wa
s not intended
    # and adds the Group.# columns for each grouping variable. The original colums are
not usable,
    # getting NA values(and warnings), unwanted aggregate values and are therefor dele
ted, the Group columns renamed
    # the Warnings are all but one removed from the file presented
motionsumar<-aggregate(phonemotion,list(phonemotion$person,phonemotion$activity),FUN=m
ean)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

```
motionsumar<-motionsumar[,-(3:4)] # removing the useless columns and subsequently rena
ming the Group.# columns
names(motionsumar)[names(motionsumar)=="Group.1"]<-"person"
names(motionsumar)[names(motionsumar)=="Group.2"]<-"activity"
write.csv(motionsumar, file =".\\motionmeans.csv")
```

Please refer to the result data file in Git for the end result