W4-Project-Xiao-Lu.R

letie

2020-05-15

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts -------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# Get the variables' names. To be on the safe side, convert them into lowercase.
labels <- read.table("features.txt") %>% select(2)
labels <- labels %>% mutate(V2 = tolower(V2))
labels <- t(labels)

# Read the X_test and set the variables' names, that is the headers.
X_test <- read.table("X_test.txt")
names(X_test) <- labels

# Read the y_test and set the headers.
y_test <- read.table("y_test.txt")
y_test <- y_test %>% rename(activity_labels = V1)

# Read the subject_test and set the headers.
subject_test <- read.table("subject_test.txt")
subject_test <- subject_test %>% rename(person_labels = V1)

# Bind the columns in the above three datasets.
test_integrated <- cbind(subject_test, y_test, X_test)

# Replicating the similar operations for the training data.
X_train <- read.table("X_train.txt")
names(X_train) <- labels

y_train <- read.table("y_train.txt")
y_train <- y_train %>% rename(activity_labels = V1)

subject_train <- read.table("subject_train.txt")
subject_train <- subject_train %>% rename(person_labels = V1)
train_integrated <- cbind(subject_train, y_train, X_train)
```

```r
# Get the large and integrated dataset.
test_and_train <- rbind(test_integrated, train_integrated)
```

```r
# Extracts only the measurements on the mean and standard deviation.
interim1 <- test_and_train[,1:2]
interim2 <- test_and_train[, grepl("mean|std", names(test_and_train))]
mean_and_std <- cbind(interim1, interim2)
```

```r
# Since I was going to use the string replacement function, i.e. str_replace_all,
# the values for activity_labels should be of class character. Given that they
# were numeric in the original dataset, I firstly used as.character to convert
# them into character.
mean_and_std[, 2] <- sapply(mean_and_std$activity_labels, as.character)
mean_and_std <- mean_and_std %>% mutate(activity_labels = str_replace_all(activity_labels,
                                                                          c("1" = "walking","2" = "walk
```

```r
averaged_data <- mean_and_std %>% group_by(activity_labels, person_labels) %>%
  summarise_at(vars(`tbodyacc-mean()-x`:`angle(z,gravitymean)`), funs(mean))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```r
write.table(averaged_data, file = "dt-step5-Xiao Lu", row.names = FALSE)
```

```r
# Below are some codes might also work, i.e. alternatives to the codes above.
# However, I didn't execute them because I prefer those above.

# This is used to remove some columns with duplicated names. Since the select()
# only works for unique names.
# test_and_train <- test_and_train[ , !duplicated(colnames(test_and_train))]

# An alternative way to construct a dataset containing only mean and std.
# mean_and_std <- test_and_train %>% select(contains("mean")|contains("std"))
```