Data-531-Lab-4

Dhun Sheth

2023-09-27

Exercise 1

```
avglist <- function(lst, low=0, high=100){
  #' @title Calculates average of a list of values
  #'
  #' @description Calculates the average of a list of values given a list and the range (low, high).
  #'The function will order the list first, and then calculate the average of all values between the lo
  #'
  #' @param lst an object of class list. List of values.
  #' @param low an object of class double. The lower value of the range, non-inclusive.
  #' @param high an object of class double. The higher value of the range, non-inclusive.
  #' @output returns either the mean of the list or an error message if list is empty, or not of type l
  #' @examples
    #' x <- list(2, 3, 10, 15, 6, 7, 7, 8, 22, 22, 23, 1)
    #' func_response <- avglist(x, 6, 22)
    #' function will return average of 9.4
    #' list its calculating the average over is (7,7,8,10,15)
  if (is.list(lst)){
    if (length(lst)==0){
      return("List is of size 0")
    }else {
      # if (length(lst) > 1){
      #   lst <- lst[order(unlist(lst))]
      # }else {
      #   lst[[1]] <- sort(unlist(lst))
      # }
      lst <- sort(unlist(lst))
      low_index <- which(unlist(lst) == low)[length(which(unlist(lst) == low))] # length(which(unlist(l
      high_index <- which(unlist(lst) == high)[1] # length(which(unlist(lst) == high)) give last high v
      if (identical(low_index, integer(0))) {
        return("Lower value not found in list")
      }else if (is.na(high_index)) {
        return("Upper value not found in list")
      }else {
        mean_lst <- mean(unlist(lst)[(low_index + 1):(high_index - 1)])
        return(mean_lst)
      }
    }
  }else {
    return("Didn't send a list")
  }
```

```
}

?avglist # printing documentation

test <- list(sample(seq(1,100,1),100))
results <- avglist(test,30,100)
print(results)
```

```
## [1] 65
```

```
test2 <- list(sample(seq(1,100,1),100, replace = TRUE)) # with replace as true, there is a chance 1 or
results2 <- avglist(test2,30,100)
print(results2)
```

```
## [1] "Lower value not found in list"
```

Exercise 2

```
lab4 <- read_csv("lab4.csv")
```

```
## Rows: 100 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): name, gender, country, email
## dbl (3): age, height, weight
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
new_df <- data.frame(
  name = c("Diego", "Kim", "Nelson"),
  age = c(22,67,11),
  gender = c("M", "F", "M"),
  height = c(129,103,100),
  weight = c(50.77,32.11,22.74),
  country = c("Mexico","North Korea","Uganda"),
  email = c("diegor@mail.tfr.mx","kimlady@gmail.com","nSekitolenko@urj.com")
)
new_df <- as_tibble(new_df)
print(new_df)
```

```
## # A tibble: 3 x 7
##   name     age gender height weight country     email
##   <chr>  <dbl> <chr>   <dbl>  <dbl> <chr>       <chr>
## 1 Diego     22 M         129   50.8 Mexico      diegor@mail.tfr.mx
## 2 Kim       67 F         103   32.1 North Korea kimlady@gmail.com
## 3 Nelson    11 M         100   22.7 Uganda      nSekitolenko@urj.com
```

```r
combined_df <- full_join(lab4,new_df)
```

```
## Joining with `by = join_by(name, age, gender, height, weight, country, email)`
```

```r
print(combined_df)
```

```
## # A tibble: 103 x 7
##    name        age gender height weight country                 email
##    <chr>     <dbl> <chr>   <dbl>  <dbl> <chr>                   <chr>
##  1 Echo         59 F         115     NA Bosnia and Herzegovina non.magna@Suspen~
##  2 Addison      57 F         101    108 Paraguay                penatibus.et@est~
##  3 MacKensie    68 F         141    294 Germany                 primis.in.faucib~
##  4 Kelsey       77 F          97    190 Serbia                  amet.luctus.vulp~
##  5 Anika        46 F         144    168 Greece                  Donec@justoeu.net
##  6 Kieran       25 M         104    289 Georgia                 Phasellus@facili~
##  7 Ursa         64 M          63    255 Saint Lucia             dis@lobortistell~
##  8 Burke        78 M         116    132 Vanuatu                 natoque.penatibu~
##  9 Aaron         8 M          87     51 Cuba                    elit.pretium@net~
## 10 Malik         5 M          72    247 Vanuatu                 adipiscing.ligul~
## # i 93 more rows
```

```r
num_averages <- colMeans(combined_df[sapply(combined_df, is.numeric)], na.rm = TRUE)
print(num_averages)
```

```
##       age    height    weight
##  52.80583 126.35922 187.02594
```

Exercise 3

```r
sensor <- read_csv("sensor.csv")
```

```
## Rows: 4320 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): timestamp, value
## dbl (2): siteid, sensorid
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
for (i in seq(1,length(sensor$timestamp),1)){
  sensor$day[i] <- as.double(substr(sensor$timestamp[i],8,9))
  sensor$time[i] <- substr(sensor$timestamp[i],45,52)
}
```

```
## Warning: Unknown or uninitialised column: `day`.
```

```
## Warning: Unknown or uninitialised column: `time`.
```

```r
print(sensor)
```

```
## # A tibble: 4,320 x 6
##    siteid sensorid timestamp                                    value   day time
##     <dbl>    <dbl> <chr>                                        <chr> <dbl> <chr>
##  1      1        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 24        1 00:0~
##  2      1        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 5         1 00:0~
##  3      1        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 60        1 00:0~
##  4      2        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 0         1 00:0~
##  5      2        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 5         1 00:0~
##  6      2        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 100       1 00:0~
##  7      3        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 36        1 00:0~
##  8      3        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 5         1 00:0~
##  9      3        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 38        1 00:0~
## 10      4        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~ 99        1 00:0~
## # i 4,310 more rows
```

```r
sensor$value <- as.double(sensor$value)
```

```
## Warning: NAs introduced by coercion
```

```r
sensors_clean <- sensor %>% filter(value <= 100) %>% filter(value >= 0)

print(sensors_clean)
```

```
## # A tibble: 3,863 x 6
##    siteid sensorid timestamp                                    value   day time
##     <dbl>    <dbl> <chr>                                        <dbl> <dbl> <chr>
##  1      1        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~    24     1 00:0~
##  2      1        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~     5     1 00:0~
##  3      1        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~    60     1 00:0~
##  4      2        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~     0     1 00:0~
##  5      2        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~     5     1 00:0~
##  6      2        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~   100     1 00:0~
##  7      3        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~    36     1 00:0~
##  8      3        2 {'Day':01,'Month':'Sep','Year':2018,'Time'~     5     1 00:0~
##  9      3        3 {'Day':01,'Month':'Sep','Year':2018,'Time'~    38     1 00:0~
## 10      4        1 {'Day':01,'Month':'Sep','Year':2018,'Time'~    99     1 00:0~
## # i 3,853 more rows
```

```r
# count_valid <- length(sensors_clean$value)
# min_reading <- min(sensors_clean$value)
# mean_reading <- mean(sensors_clean$value)
# range_reading <- range(sensors_clean$value)
# max_reading_site_2 <- max(filter(sensors_clean, siteid == 2)$value)
# count_site_1_sensor_2 <- length(filter(sensors_clean, siteid == 1 & sensorid == 2)$value)

# assume summary is for sensor dataframe
count_valid <- length(sensors_clean$value)
min_reading <- min(filter(sensor, !is.na(value))$value)
mean_reading <- mean(filter(sensor, !is.na(value))$value)
```

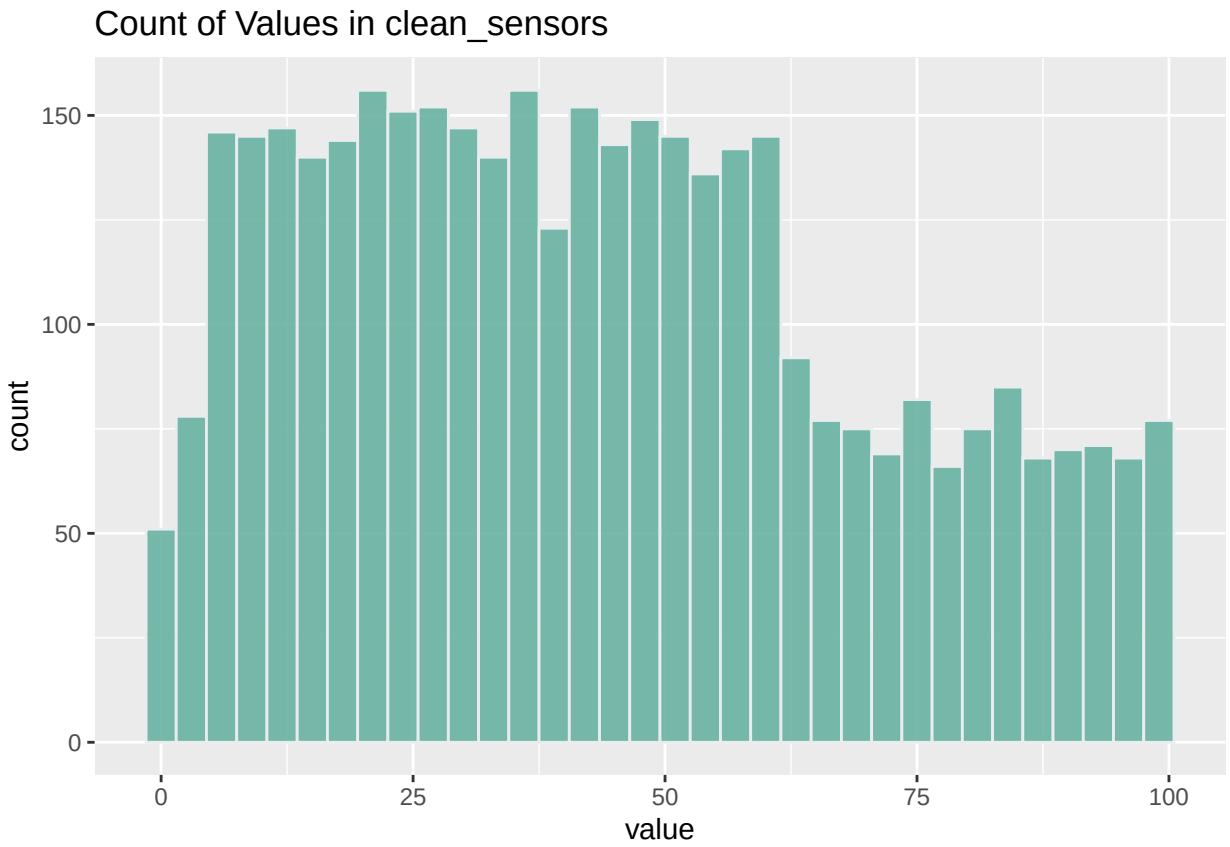
```r
range_reading <- range(filter(sensor, !is.na(value))$value)
max_reading_site_2 <- max(filter(sensor, !is.na(value) & siteid == 2)$value)
count_site_1_sensor_2 <- length(filter(sensor, siteid == 1 & sensorid == 2)$value)


data_summary <- list(count_valid, min_reading, mean_reading, range_reading,
                     max_reading_site_2, count_site_1_sensor_2)
names(data_summary) <- c("count_valid","min_reading","mean_reading", "range_reading",
                         "max_reading_site_2", "count_site_1_sensor_2")
print(data_summary)
```

```
## $count_valid
## [1] 3863
##
## $min_reading
## [1] -99
##
## $mean_reading
## [1] 43.65486
##
## $range_reading
## [1] -99 199
##
## $max_reading_site_2
## [1] 187
##
## $count_site_1_sensor_2
## [1] 288
```

```r
sensors_clean_hist <- ggplot(sensors_clean, aes(x=value)) +
  geom_histogram( binwidth=3, fill="#69b3a2", color="#e9ecef", alpha=0.9) +
  ggtitle("Count of Values in clean_sensors")

print(sensors_clean_hist)
```

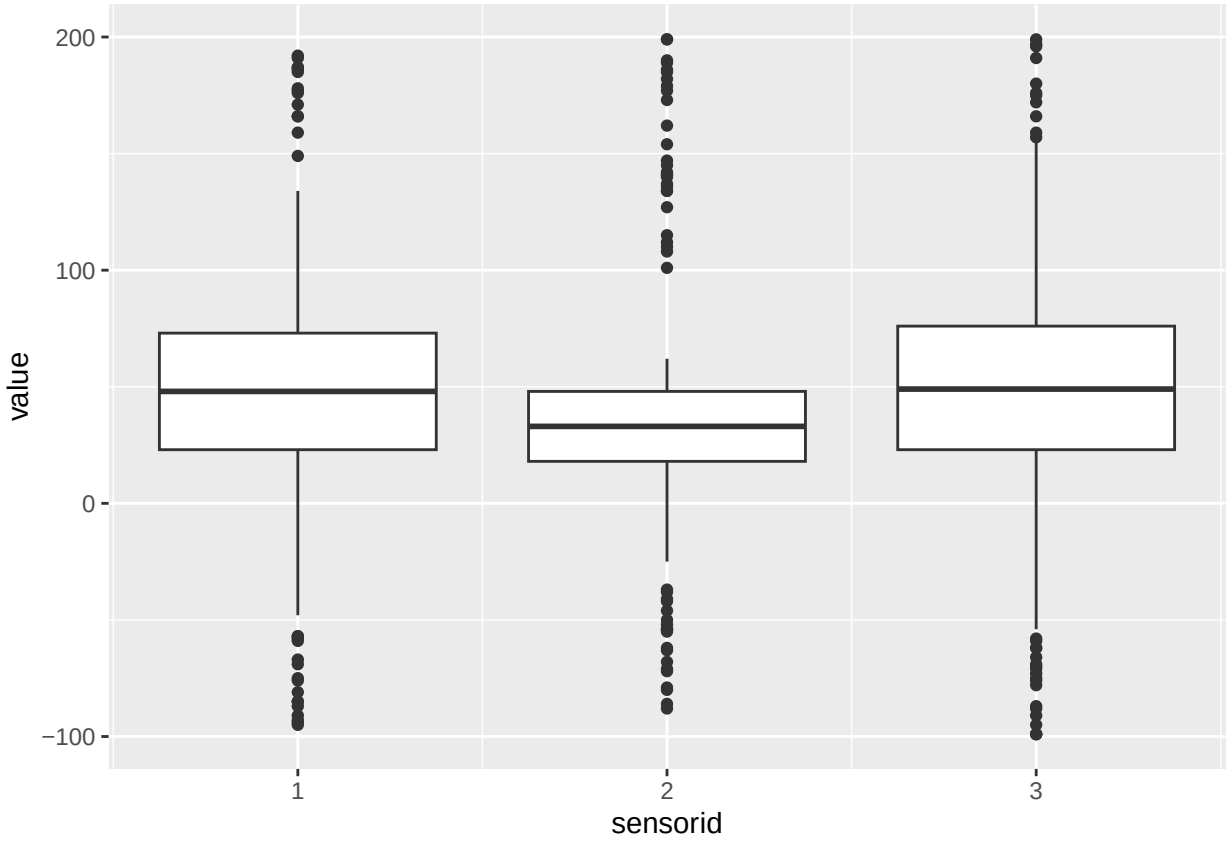



```
# assume box plot is for the total sensor data
# sensors_clean_box_plot <- ggplot(sensors_clean, aes(x=sensorid, y=value, group=sensorid)) +
#    geom_boxplot()

sensor_box_plot <- ggplot(sensor, aes(x=sensorid, y=value, group=sensorid)) +
   geom_boxplot()

print(sensor_box_plot)
```

```
## Warning: Removed 284 rows containing non-finite values (‘stat_boxplot()‘).
```


200
100
0
−100
value
1
2
3
sensorid