Liotta_Dario_Rlab01

2024-04-09

Libraries used for this exercise:

```
library(dplyr)
```

```
##
## 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
```

```
library(ggplot2)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(geosphere)
library(RColorBrewer)
```

1. read the data and import in a `data.frame` or `tibble` structure

```
PATH     <- "/Users/darioliotta/Documents/Physics\ of\ Data/Advance\ Statistics\ for\ Physics/Labs/Lab1/
BASENAME <- "JC-20190"
LASTNAME <- "-citibike-tripdata.csv"

n <- 2

file_name <- c()
dataset   <- list()

for (i in 1:5) {

  file_name[i] <- sprintf("%s%s%i%s", PATH, BASENAME, n, LASTNAME)
  dataset[[i]] <- read.csv(file_name[i])
  n <- n + 1

}
```

2. merge the five data frames in an unique structure

```
dataset <- bind_rows(dataset)
head(dataset, 10)
```

```
##    tripduration               starttime                stoptime
## 1           142 2019-02-01 15:35:02.0820 2019-02-01 15:37:24.1360
## 2           223 2019-02-01 17:00:46.8900 2019-02-01 17:04:30.5500
## 3           106 2019-02-01 17:08:01.3260 2019-02-01 17:09:47.4400
## 4           370 2019-02-01 17:09:31.2100 2019-02-01 17:15:41.6550
## 5           315 2019-02-01 17:19:53.2490 2019-02-01 17:25:09.1400
## 6           145 2019-02-01 17:32:53.2630 2019-02-01 17:35:18.7510
## 7          1625 2019-02-01 17:39:55.5390 2019-02-01 18:07:01.2660
## 8           485 2019-02-01 17:46:32.7540 2019-02-01 17:54:37.9890
## 9           632 2019-02-01 18:05:26.8150 2019-02-01 18:15:59.1050
## 10          235 2019-02-01 18:11:06.8410 2019-02-01 18:15:02.7810
##    start.station.id start.station.name start.station.latitude
## 1              3183     Exchange Place               40.71625
## 2              3183     Exchange Place               40.71625
## 3              3183     Exchange Place               40.71625
## 4              3183     Exchange Place               40.71625
## 5              3183     Exchange Place               40.71625
## 6              3183     Exchange Place               40.71625
## 7              3183     Exchange Place               40.71625
## 8              3183     Exchange Place               40.71625
## 9              3183     Exchange Place               40.71625
## 10             3183     Exchange Place               40.71625
##    start.station.longitude end.station.id end.station.name end.station.latitude
## 1                -74.03346           3639       Harborside              40.71925
## 2                -74.03346           3681         Grand St              40.71518
## 3                -74.03346           3184      Paulus Hook              40.71415
## 4                -74.03346           3211       Newark Ave              40.72153
## 5                -74.03346           3273     Manila & 1st              40.72165
## 6                -74.03346           3214 Essex Light Rail              40.71277
## 7                -74.03346           3186    Grove St PATH              40.71959
## 8                -74.03346           3186    Grove St PATH              40.71959
## 9                -74.03346           3278 Monmouth and 6th              40.72569
## 10               -74.03346           3681         Grand St              40.71518
##    end.station.longitude bikeid   usertype birth.year gender
## 1              -74.03423  29677 Subscriber       1963      1
## 2              -74.03768  26234 Subscriber       1992      2
## 3              -74.03355  29588 Subscriber       1960      1
## 4              -74.04630  29250 Subscriber       1976      1
## 5              -74.04288  29586 Subscriber       1980      1
## 6              -74.03649  26153 Subscriber       1984      1
## 7              -74.04312  29242 Subscriber       1993      1
## 8              -74.04312  29496 Subscriber       1970      1
## 9              -74.04879  26307   Customer       1981      1
## 10             -74.03768  26293 Subscriber       1982      1
```

3. check for missing data and remove it, if any

```
dataset2 <- na.omit(dataset)
```

4.1 compute the average and the median trip duration in minutes

```
duration_mean   <- mean(dataset$tripduration)   / 60.00
duration_median <- median(dataset$tripduration) / 60.00

print(sprintf("Average trip duration: %.2f minutes", duration_mean))
```

```
## [1] "Average trip duration: 12.81 minutes"
```

```
print(sprintf("Median trip duration: %.2f minutes", duration_median))
```

```
## [1] "Median trip duration: 5.68 minutes"
```

4.2 evaluate the minimum and maximum trip duration; does that sound like a reasonable value?

```
duration_min <- min(dataset$tripduration) / 60.00
duration_max <- max(dataset$tripduration) / 60.00

print(sprintf("Minimum trip duration: %.2f minutes", duration_min))
```

```
## [1] "Minimum trip duration: 1.02 minutes"
```

```
print(sprintf("Maximum trip duration: %.2f minutes", duration_max))
```

```
## [1] "Maximum trip duration: 28817.00 minutes"
```

4.3 repeat the calculation of the average (and the median) trip duration by excluding trips longer than 3 hours. Next, evaluate the number of skimmed entries

```
dataset_filtered <- dataset[dataset$tripduration <= (60 * 3 * 60),]

duration_mean_filtered   <- mean(dataset_filtered$tripduration)   / 60.00
duration_median_filtered <- median(dataset_filtered$tripduration) / 60.00

print(sprintf("Average trip duration with skimmed data: %.2f minutes", duration_mean_filtered))
```

```
## [1] "Average trip duration with skimmed data: 9.22 minutes"
```

```
print(sprintf("Median trip duration with skimmed data: %.2f minutes", duration_median_filtered))
```

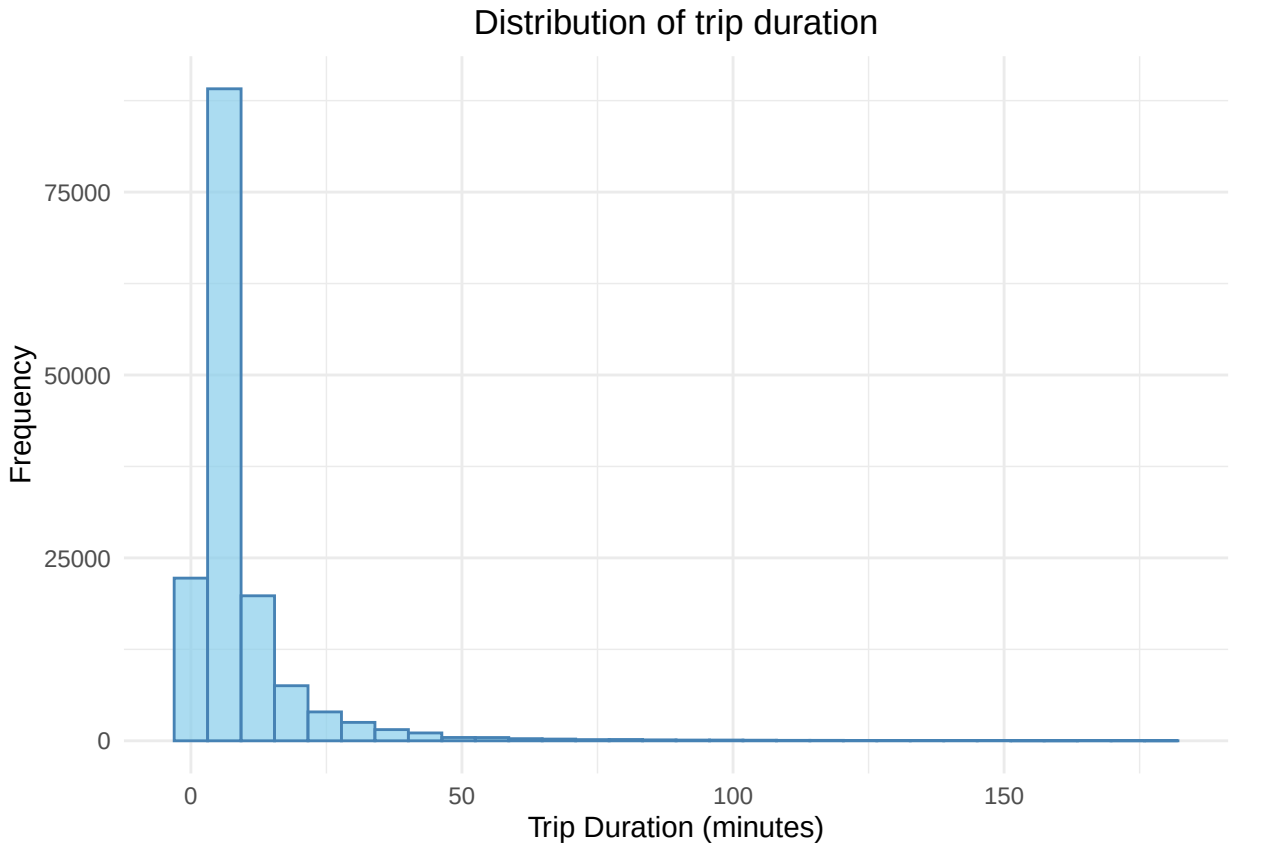
```
## [1] "Median trip duration with skimmed data: 5.67 minutes"
```

```
entries_skimmed <- nrow(dataset) - nrow(dataset_filtered)
print(sprintf("Number of entries excluded (trips duration over 3 hours): %d", entries_skimmed))
```

```
## [1] "Number of entries excluded (trips duration over 3 hours): 427"
```

4.4 plot the distribution of trip duration after the skimming of the previous point

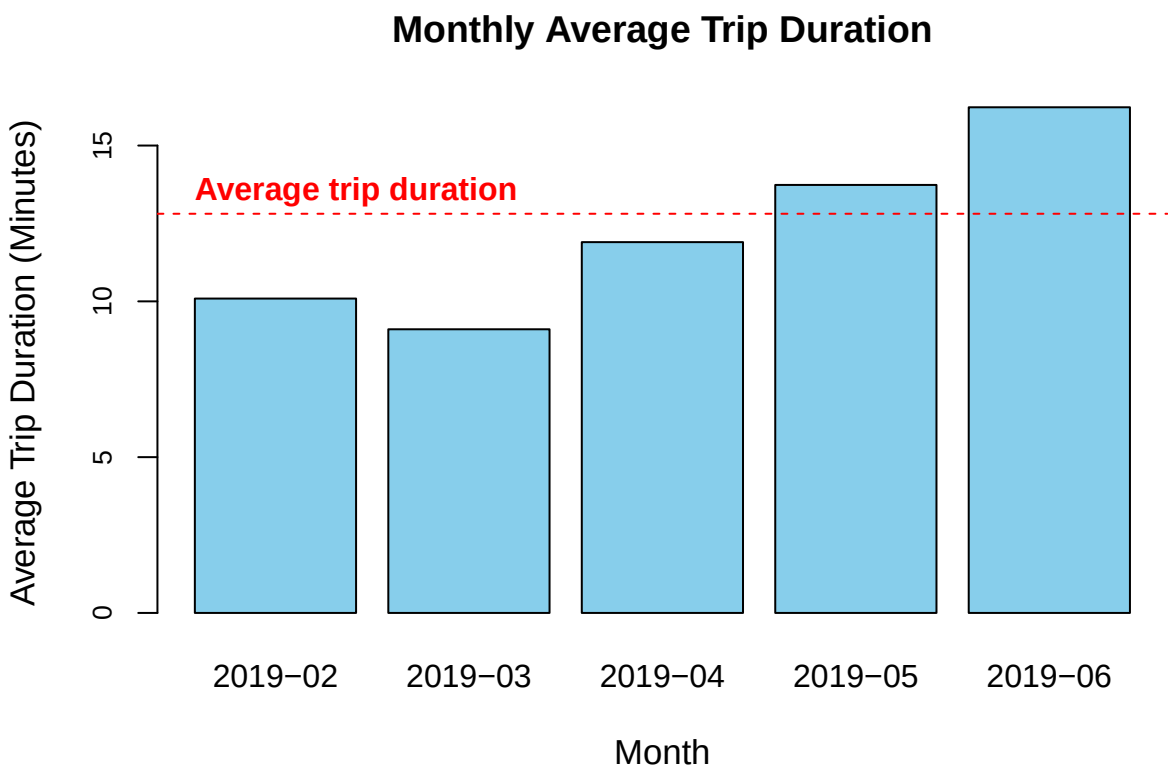
```
ggplot(dataset_filtered, aes(x = tripduration / 60.00)) +
  geom_histogram(bins = 30, color = "steelblue", fill = "skyblue", alpha = 0.7) +
  labs(title = "Distribution of trip duration",
           x = "Trip Duration (minutes)",
           y = "Frequency") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



5. plot the monthly average trip duration

```
times   <- dataset$tripduration
months  <- dataset$starttime

for (i in 1:length(months)) {
  months[i] <- substr(months[i], 1, 7)
}

data <- tapply(times, months, mean)
monthly_average_trip_duration <- double(length(data))

for (i in 1:length(data)) {
  monthly_average_trip_duration[i] <- data[[i]] / 60.00
```

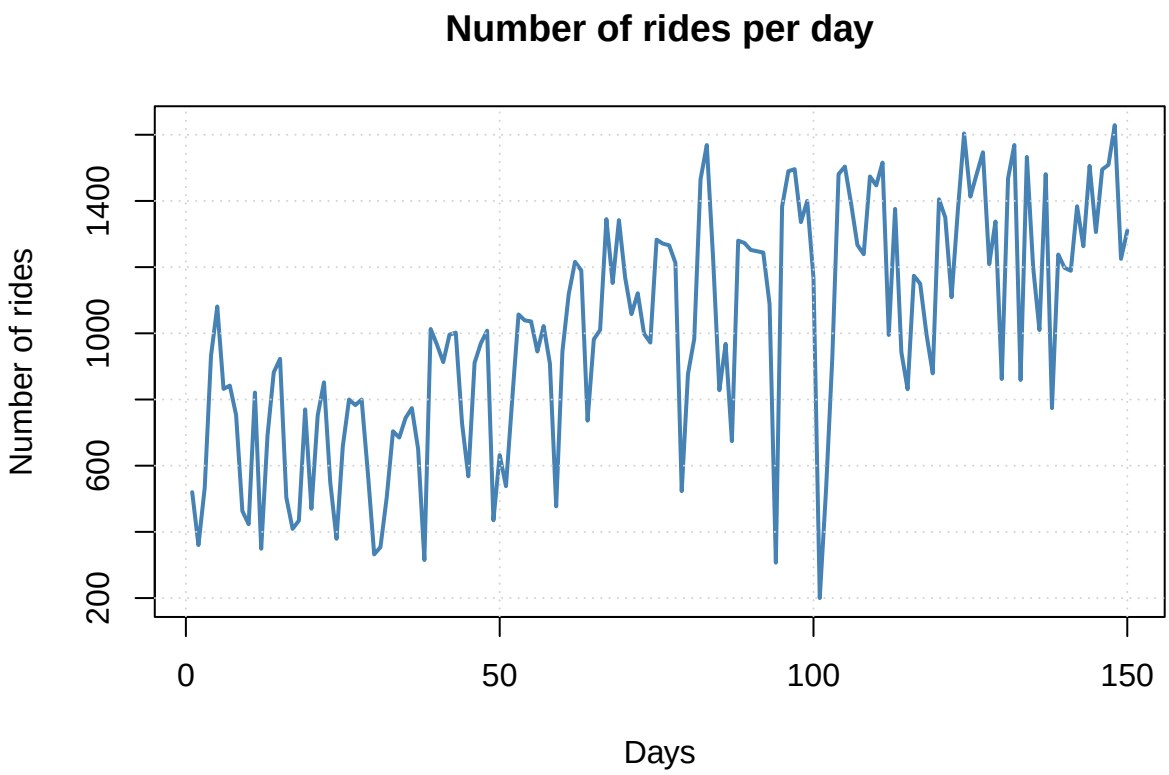
```
}

barplot(monthly_average_trip_duration,
        names.arg = names(data),
        main      = "Monthly Average Trip Duration",
        xlab      = "Month",
        ylab      = "Average Trip Duration (Minutes)",
        col       = "skyblue",
        border    = "black",
        cex.axis  = 0.8,
        cex.lab   = 1.1,
)

abline(h = duration_mean, col = "red", lty = 2)
text(x = 1.2, y = 13.5, labels = "Average trip duration", col = "red", font = 2)
```



6.1 plot the number of rides per day

```
times     <- dataset$tripduration
startdays <- dataset$starttime

for (i in 1:length(startdays)) {
  startdays[i] <- substr(startdays[i], 1, 10)
}

data <- tapply(startdays, startdays, length)
number_of_rides_per_day <- c(length(data))
```

```r
for (i in 1:length(data)) {
  number_of_rides_per_day[i] <- data[[i]]
}

plot(1:length(number_of_rides_per_day),
     number_of_rides_per_day,
     type = "l",
     main = "Number of rides per day",
     xlab = "Days",
     ylab = "Number of rides",
     col  = "steelblue",
     lwd  = 2
)
grid()
```

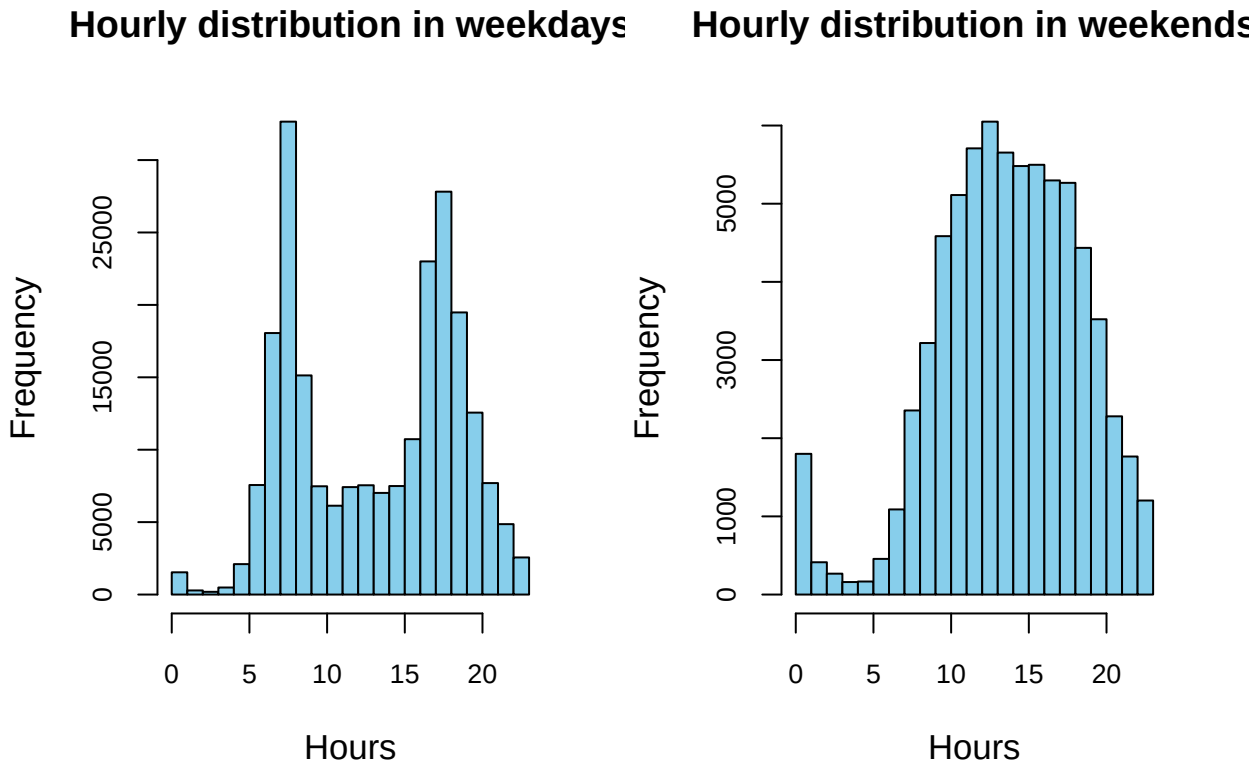


6.2 plot the hourly distribution on weekdays and on weekends

```r
starthours <- dataset$starttime

for (i in 1:length(starthours)) {
  starthours[i] <- substr(starthours[i], 12, 13)
}

startdays         <- as.POSIXct(startdays)
startdays_of_week <- weekdays(startdays)

data <- dataset$stoptime
```

```
stopdays  <- character(length(data))
stophours <- character(length(data))

for (i in 1:length(data)) {
  stopdays[i]  <- substr(data[[i]], 1, 10)
  stophours[i] <- substr(data[[i]], 12, 13)
}

stopdays         <- as.POSIXct(stopdays)
stopdays_of_week <- weekdays(stopdays)

weekdays = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
weekends = c("Saturday", "Sunday")

startmask = logical(length(startdays))
stopmask  = logical(length(stopdays))

for (i in 1:length(startdays_of_week)) {
  if (startdays_of_week[i] %in% weekdays) {
    startmask[i] <- TRUE
  }
  else if (startdays_of_week[i] %in% weekends) {
    startmask[i] <- FALSE
  }
}

for (i in 1:length(stopdays_of_week)) {
  if (stopdays_of_week[i] %in% weekdays) {
    stopmask[i] <- TRUE
  }
  else if (stopdays_of_week[i] %in% weekends) {
    stopmask[i] <- FALSE
  }
}

starthours_weekdays <- as.numeric(ifelse(!startmask, -1, starthours))
starthours_weekends <- as.numeric(ifelse(startmask, -1, starthours))
stophours_weekdays  <- as.numeric(ifelse(!stopmask, -1, stophours))
stophours_weekends  <- as.numeric(ifelse(stopmask, -1, stophours))

par(mfrow = c(1, 2))

data <- c(starthours_weekdays, stophours_weekdays)

hist(data[data != -1],
     breaks   = 24,
     xlab     = "Hours",
     ylab     = "Frequency",
     main     = "Hourly distribution in weekdays",
     col      = "skyblue",
     border   = "black",
     cex.axis = 0.8,
     cex.lab  = 1.1
```

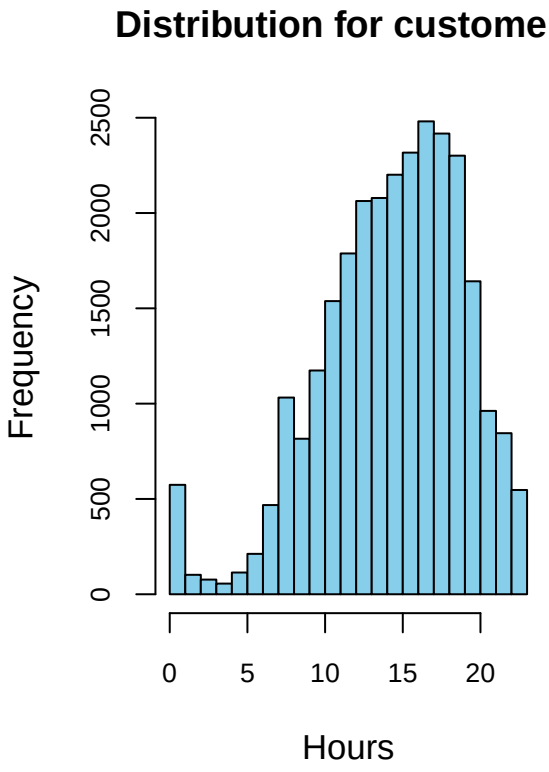
```
)

data <- c(starthours_weekends, stophours_weekends)

hist(data[data != -1],
     breaks   = 24,
     xlab     = "Hours",
     ylab     = "Frequency",
     main     = "Hourly distribution in weekends",
     col      = "skyblue",
     border   = "black",
     cex.axis = 0.8,
     cex.lab  = 1.1,
)
```

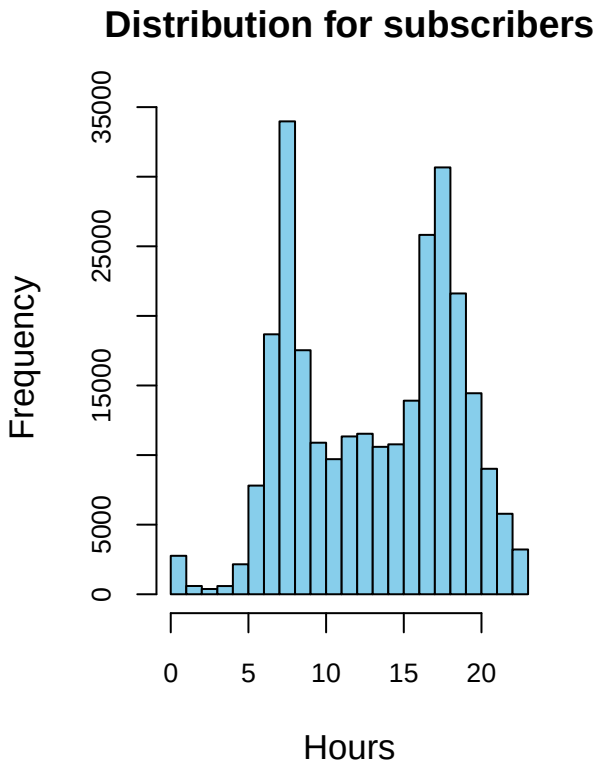


6.3 plot again the average hourly distribution on weekdays but separating customer and subscriber users

```
usertype = dataset$usertype
usermask = logical(length(usertype))

for (i in 1:length(usertype)) {
  if (usertype[i] == "Customer") {
    usermask[i] <- TRUE
  }
  else if (usertype[i] == "Subscriber") {
    usermask[i] <- FALSE
```

```
  }
}

starthours_weekdays_customer     <- as.numeric(ifelse(!usermask, -1, starthours))
stophours_weekdays_customer      <- as.numeric(ifelse(!usermask, -1, stophours))
starthours_weekdays_subscriber   <- as.numeric(ifelse(usermask, -1, starthours))
stophours_weekdays_subscriber    <- as.numeric(ifelse(usermask, -1, stophours))

par(mfrow = c(1, 2))

data <- c(starthours_weekdays_customer, stophours_weekdays_customer)

hist(data[data != -1],
     breaks   = 24,
     xlab     = "Hours",
     ylab     = "Frequency",
     main     = "Distribution for customers",
     col      = "skyblue",
     border   = "black",
     cex.axis = 0.8,
     cex.lab  = 1.1
)

data <- c(starthours_weekdays_subscriber, stophours_weekdays_subscriber)

hist(data[data != -1],
     breaks   = 24,
     xlab     = "Hours",
     ylab     = "Frequency",
     main     = "Distribution for subscribers",
     col      = "skyblue",
     border   = "black",
     cex.axis = 0.8,
     cex.lab  = 1.1,
)
```





7.1 using the latitude and longitude information, evaluate the average speed (in km/h) of a user, discarding the trip lasting longer than 1 hour

```
x_start <- dataset$start.station.latitude
y_start <- dataset$start.station.longitude
x_stop  <- dataset$end.station.latitude
y_stop  <- dataset$end.station.longitude

distances <- double(length(x_start))

for (i in 1:length(x_start)) {
  distances[i] <- distHaversine(c(x_start[i], y_start[i]), c(x_stop[i], y_stop[i])) / 1000.00
}

time_in_hour <- dataset$tripduration / 60.00 / 60.00

average_speed <- distances / time_in_hour
average_speed <- average_speed[time_in_hour < 1]

print("Average speed of 100 users (km/h):")
```

```
## [1] "Average speed of 100 users (km/h):"
```

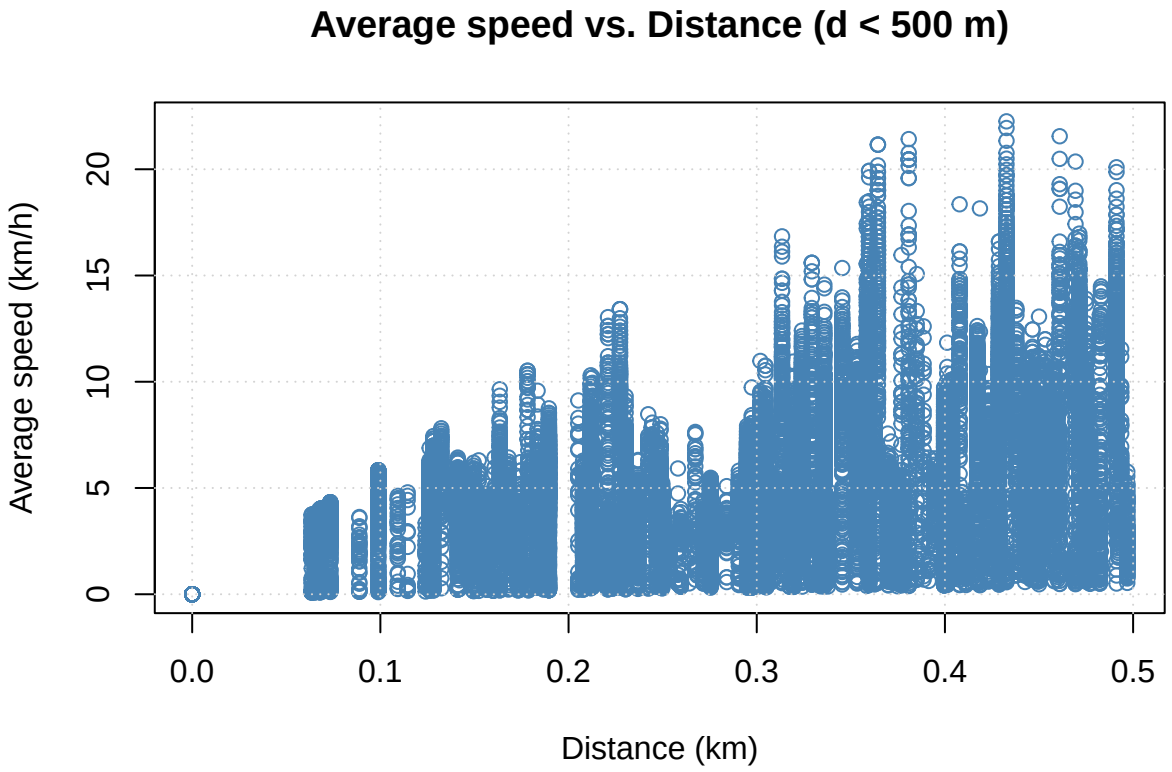
```
head(average_speed, 100)
```

```
##   [1]  3.1980276  7.6101767  2.2136514 14.0018556 12.1392182  8.7723673
##   [7]  2.3927168  8.0168345  9.8599869  7.2215719  9.9421328  9.0033128
```

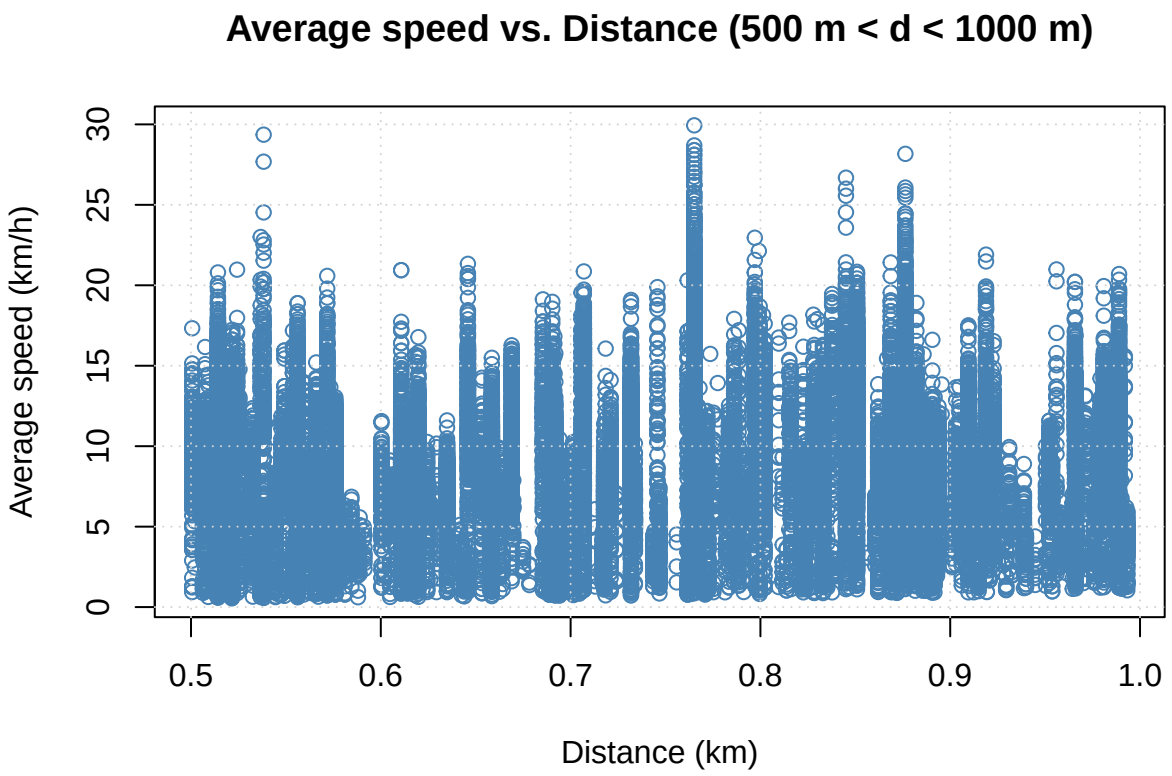
```
##  [13]  8.2637817  8.5696865  8.4799551  2.8147581  9.2470901  7.3795981
##  [19]  7.3023385  4.1283630  9.4696992  5.7483535  1.7340001  8.3193288
##  [25]  8.0907783  6.5482997 13.3666289  2.3354326 11.7359170 12.3998694
##  [31] 10.5085533  1.7174795  1.7030469  0.9520334 11.4357786  2.9330881
##  [37]  2.6364837 20.1231395 13.0098434 13.4064849  0.0000000 13.9984493
##  [43] 15.3448779  7.7382990  5.1206522  9.3150277  7.0446480 11.0665662
##  [49]  6.6409852  7.5184210  2.0765225  7.9551758  6.8899140  4.3665377
##  [55]  3.4854859  6.8386734 19.3441031  7.4599119  2.9139478  5.7921370
##  [61]  9.6482130  9.5297136  9.4443220 11.7833189  7.8259024  3.4186732
##  [67]  2.5227707 15.2114632  4.5920334 11.7011256  9.0212288 12.8322268
##  [73]  8.2954565  3.2819787  9.5879627  3.8349365  4.9666379  5.6533034
##  [79]  5.0475923  3.0082955 17.0271989  7.7382990  7.0300479 15.1357843
##  [85] 17.4486820  7.6637375  2.3232381 11.8822702 13.1286622  6.7301231
##  [91]  8.0664620 13.7390980 10.3245305  9.0844970  7.2685329 13.3148844
##  [97]  5.7556256  9.0398434  3.5203095 15.9897733
```

7.2 plot the average speed as a function of route length for the following group of distances d < 500 m, 500m < d < 1000m, 1000m < d < 2000m, 2000m < d < 3000m, d > 3000m and discarding trips longer than 1 hour

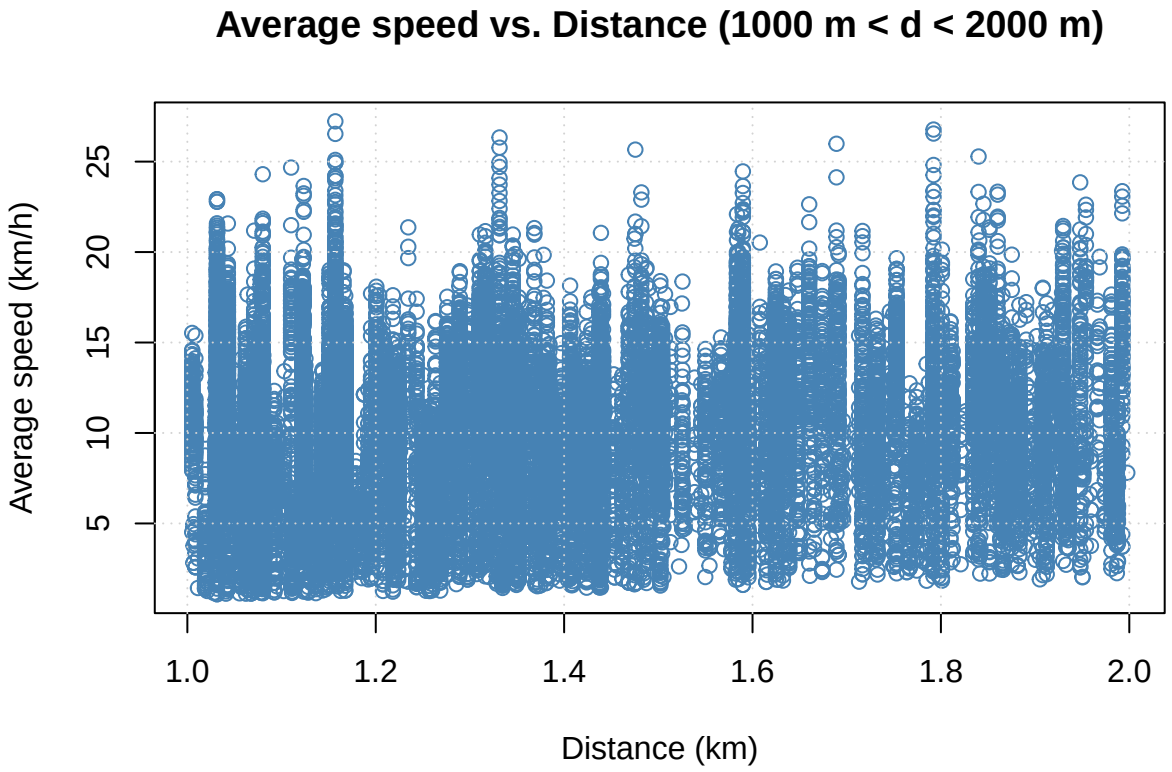
```r
distances <- distances[time_in_hour < 1]

plot(distances[distances < 0.5],
     average_speed[distances < 0.5],
     main = "Average speed vs. Distance (d < 500 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



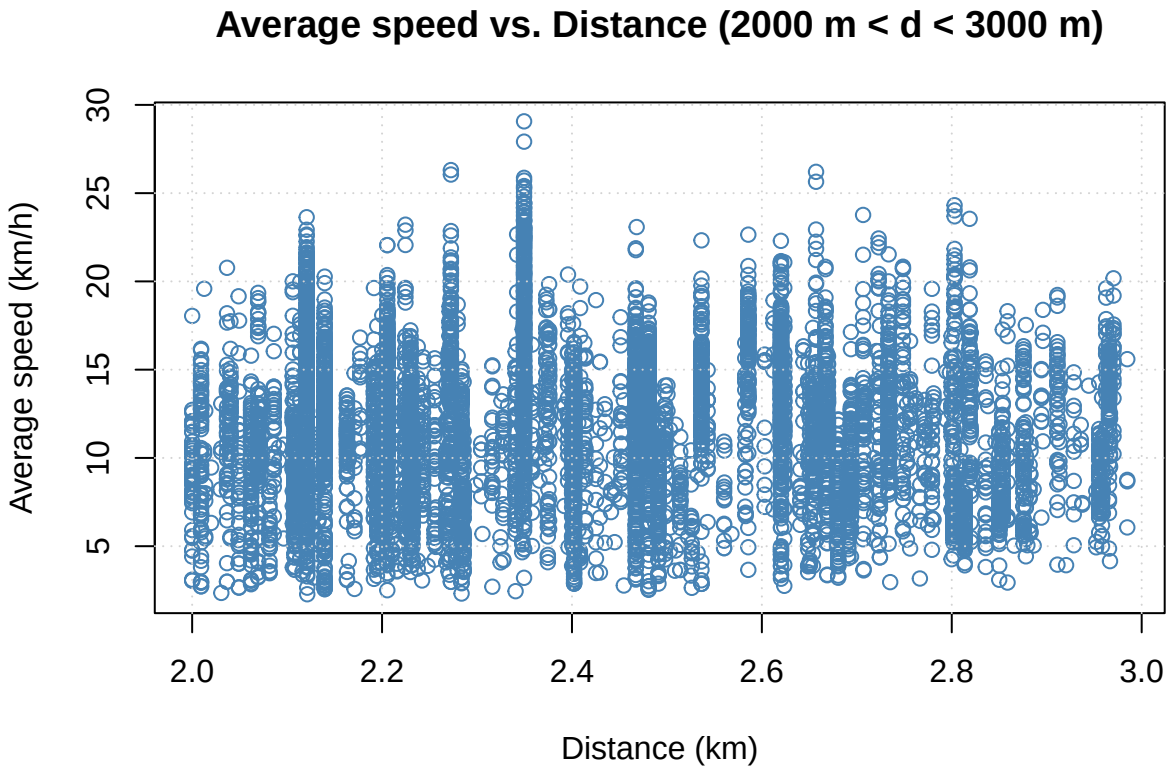

```
plot(distances[distances > 0.5 & distances < 1.0],
     average_speed[distances > 0.5 & distances < 1.0],
     main = "Average speed vs. Distance (500 m < d < 1000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



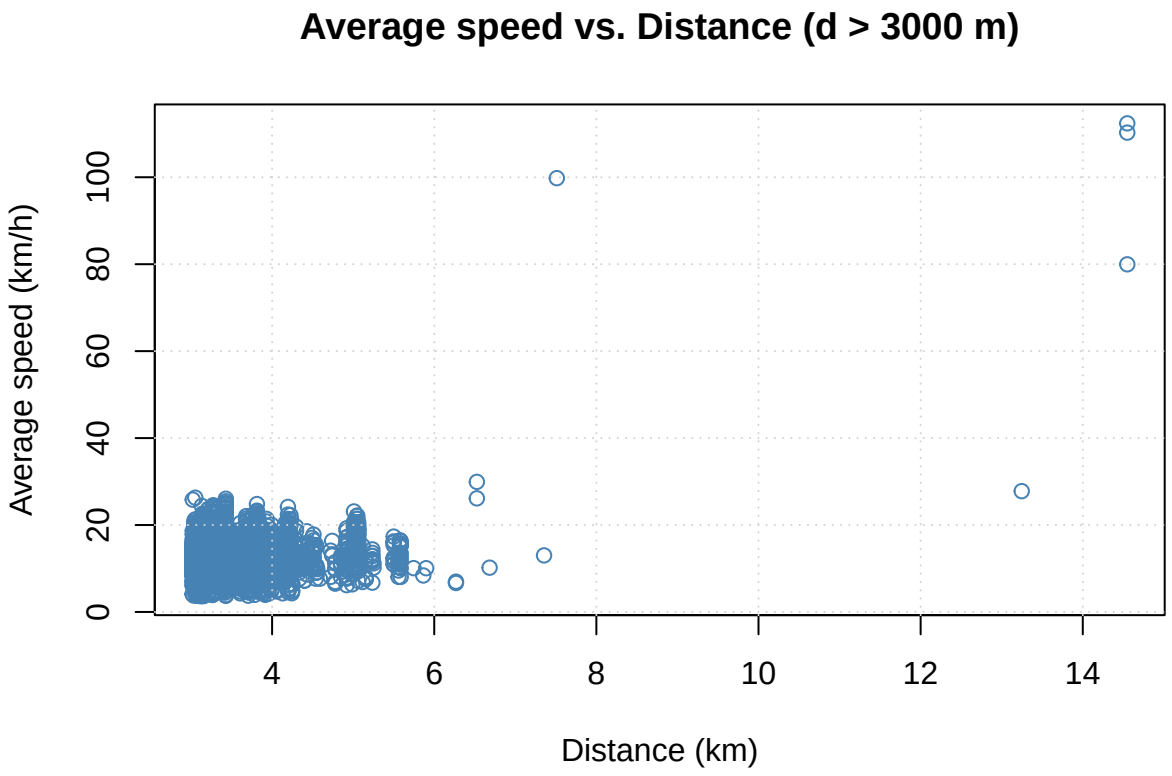

```r
plot(distances[distances > 1.0 & distances < 2.0],
     average_speed[distances > 1.0 & distances < 2.0],
     main = "Average speed vs. Distance (1000 m < d < 2000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```




```
plot(distances[distances > 2.0 & distances < 3.0],
     average_speed[distances > 2.0 & distances < 3.0],
     main = "Average speed vs. Distance (2000 m < d < 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```

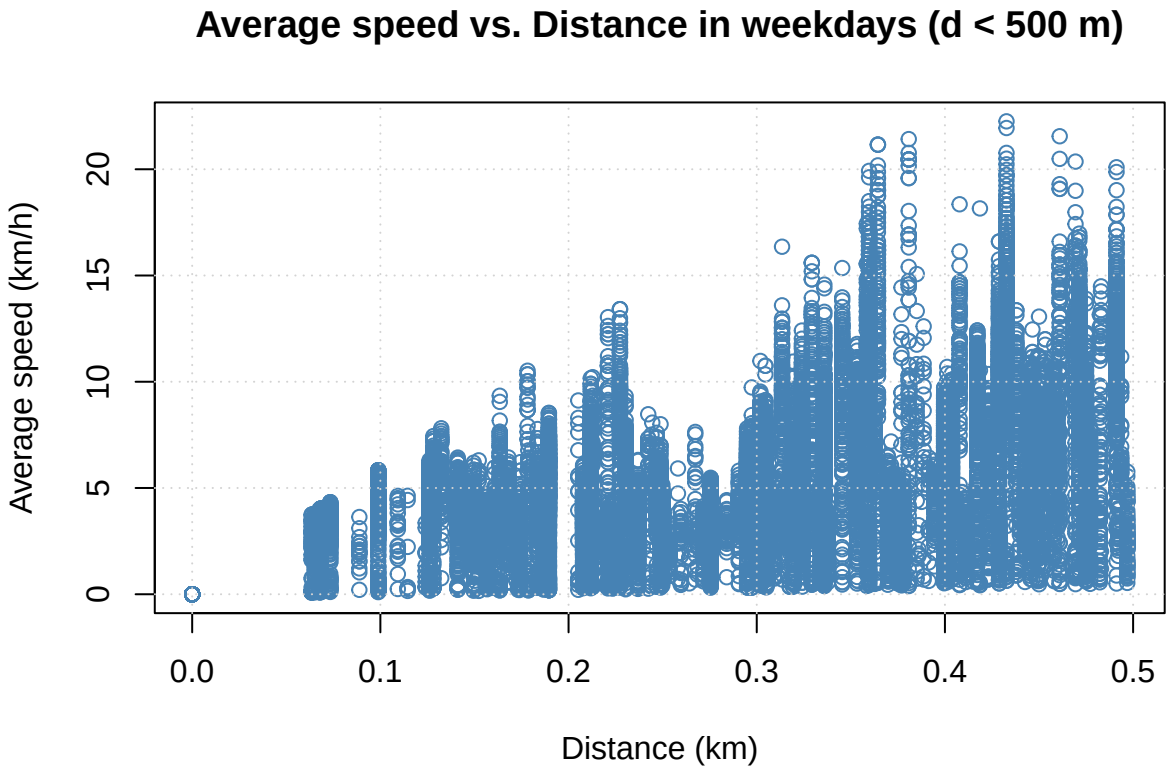



```
plot(distances[distances > 3.0],
     average_speed[distances > 3.0],
     main = "Average speed vs. Distance (d > 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```

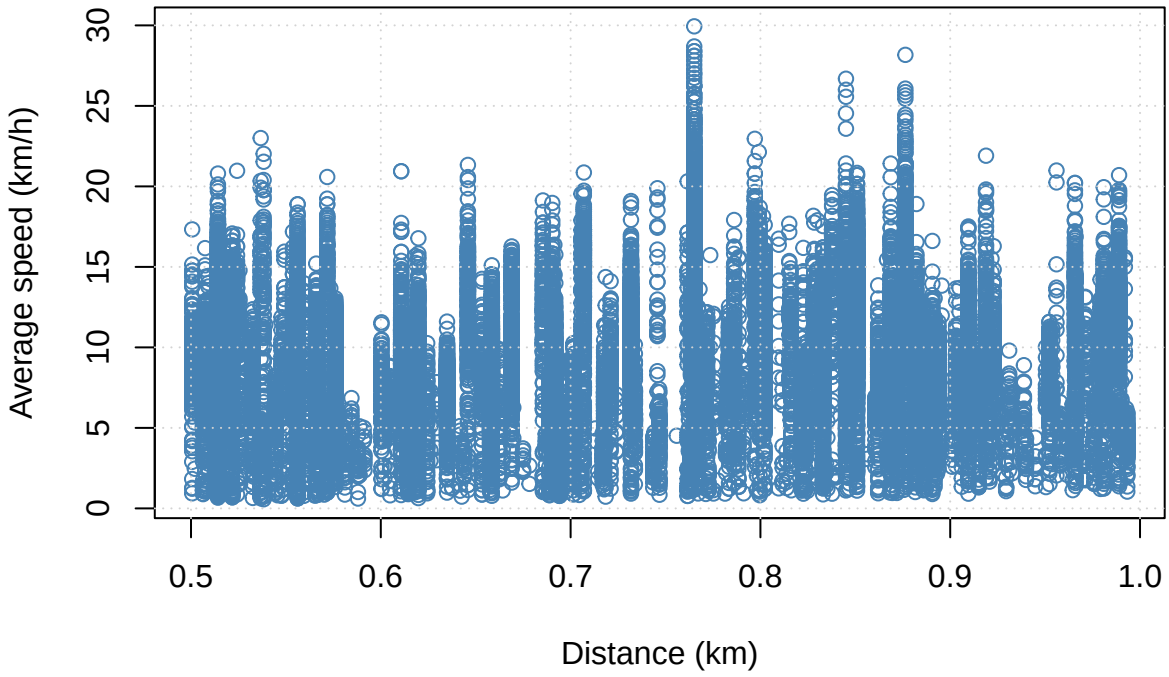


7.3 repeat the same graph, but show the results obtained separately for weekdays and weekends

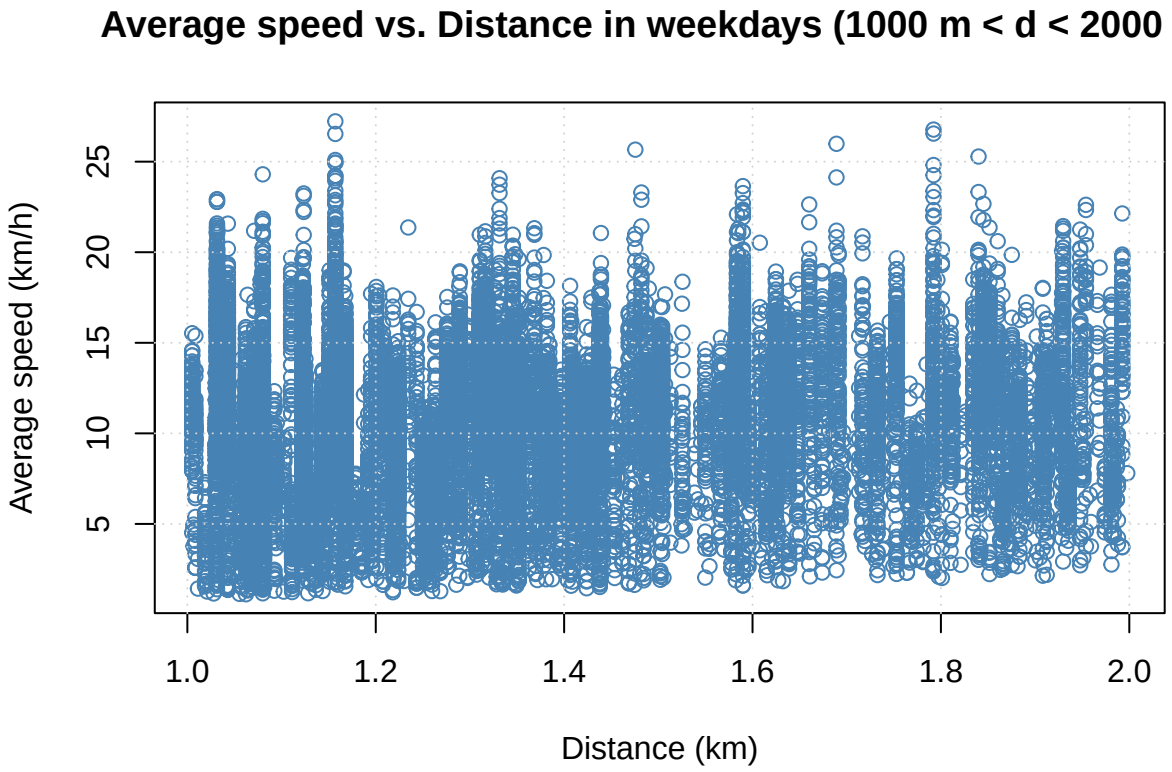
```
startmask = startmask[time_in_hour < 1]

plot(distances[distances < 0.5 & startmask],
     average_speed[distances < 0.5 & startmask],
     main = "Average speed vs. Distance in weekdays (d < 500 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



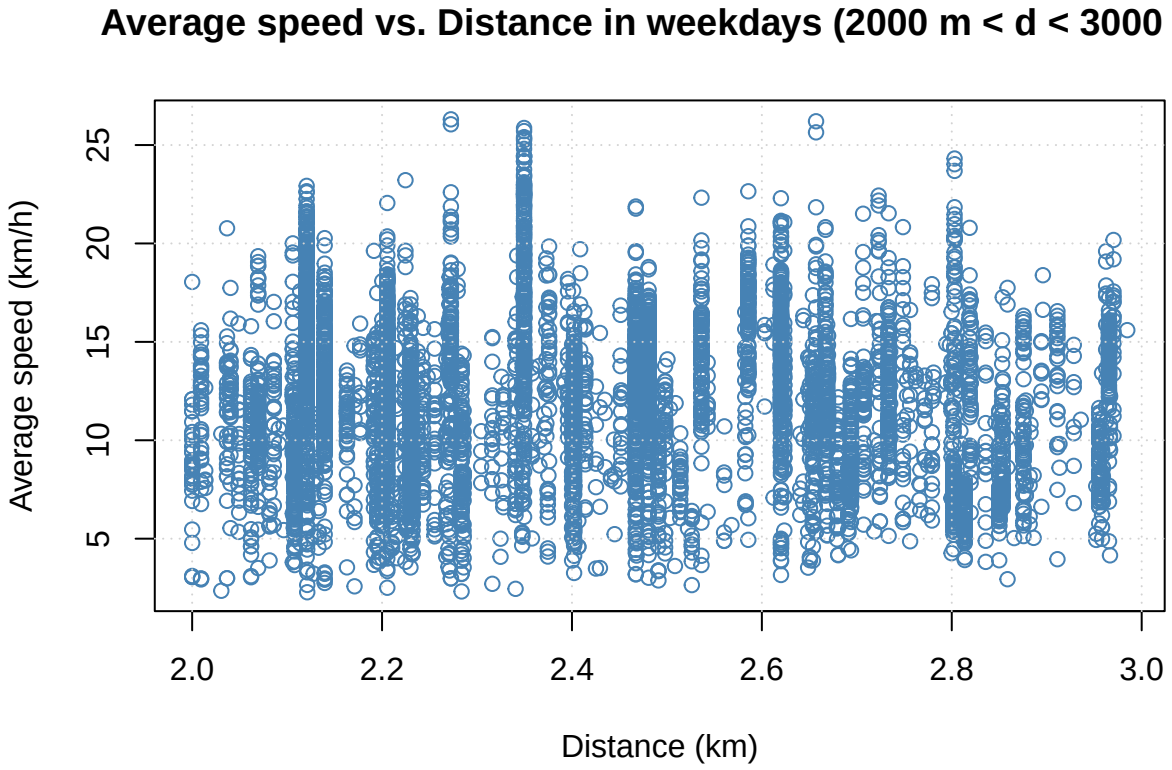

```
plot(distances[distances > 0.5 & distances < 1.0 & startmask],
     average_speed[distances > 0.5 & distances < 1.0 & startmask],
     main = "Average speed vs. Distance in weekdays (500 m < d < 1000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



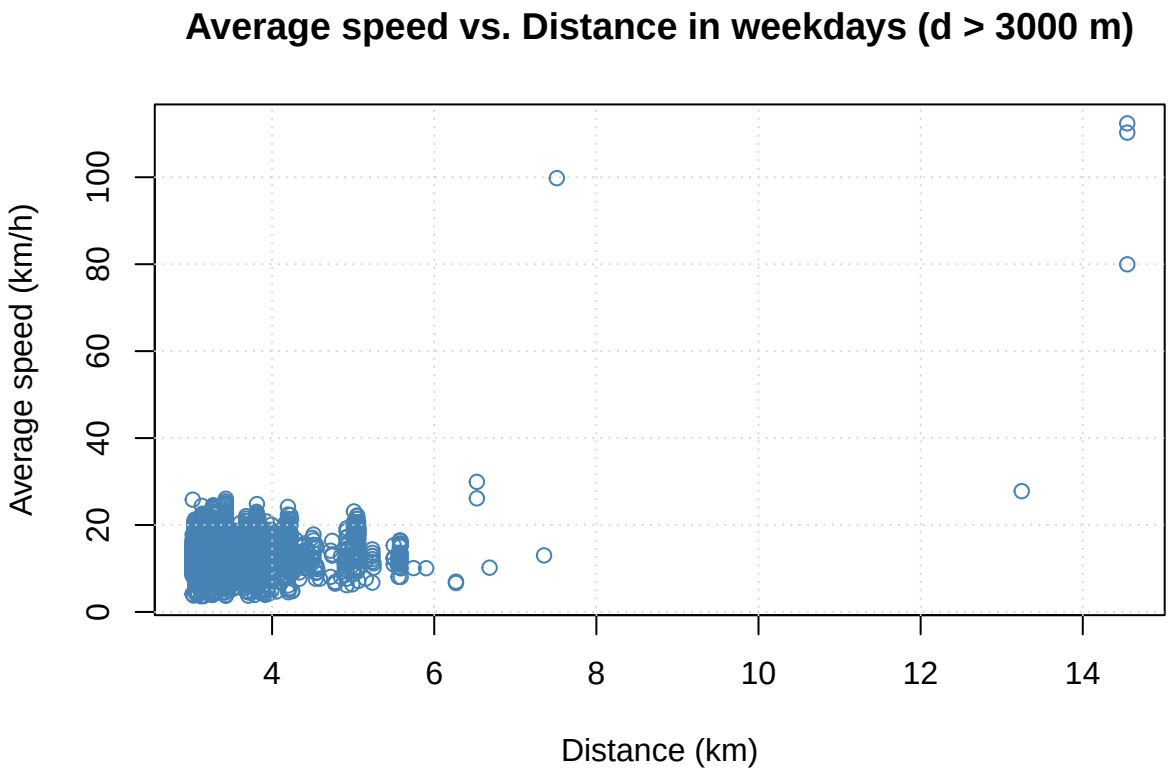

```r
plot(distances[distances > 1.0 & distances < 2.0 & startmask],
     average_speed[distances > 1.0 & distances < 2.0 & startmask],
     main = "Average speed vs. Distance in weekdays (1000 m < d < 2000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



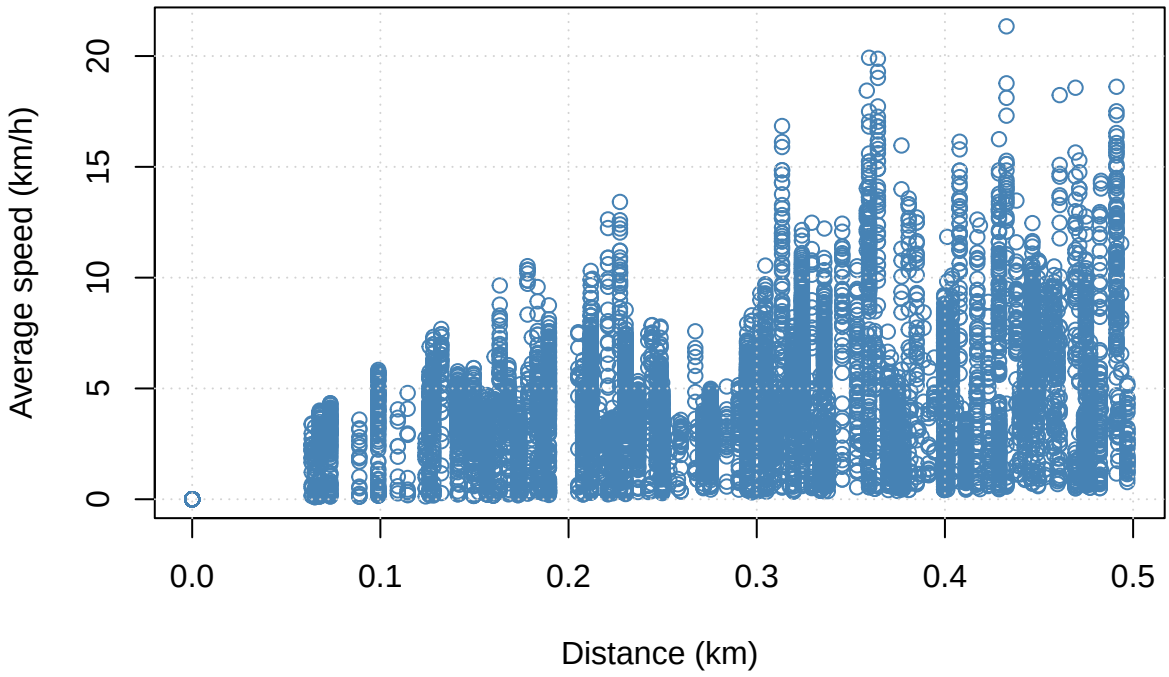

```
plot(distances[distances > 2.0 & distances < 3.0 & startmask],
     average_speed[distances > 2.0 & distances < 3.0 & startmask],
     main = "Average speed vs. Distance in weekdays (2000 m < d < 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



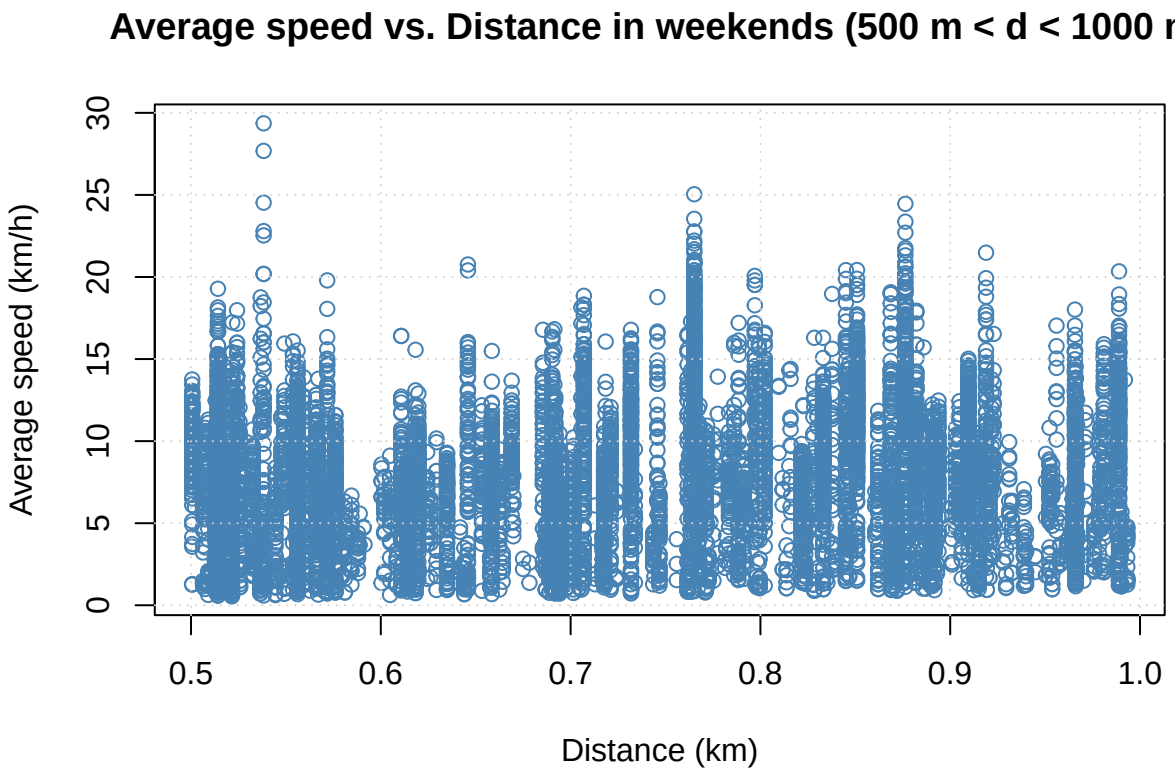

```
plot(distances[distances > 3.0 & startmask],
     average_speed[distances > 3.0 & startmask],
     main = "Average speed vs. Distance in weekdays (d > 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



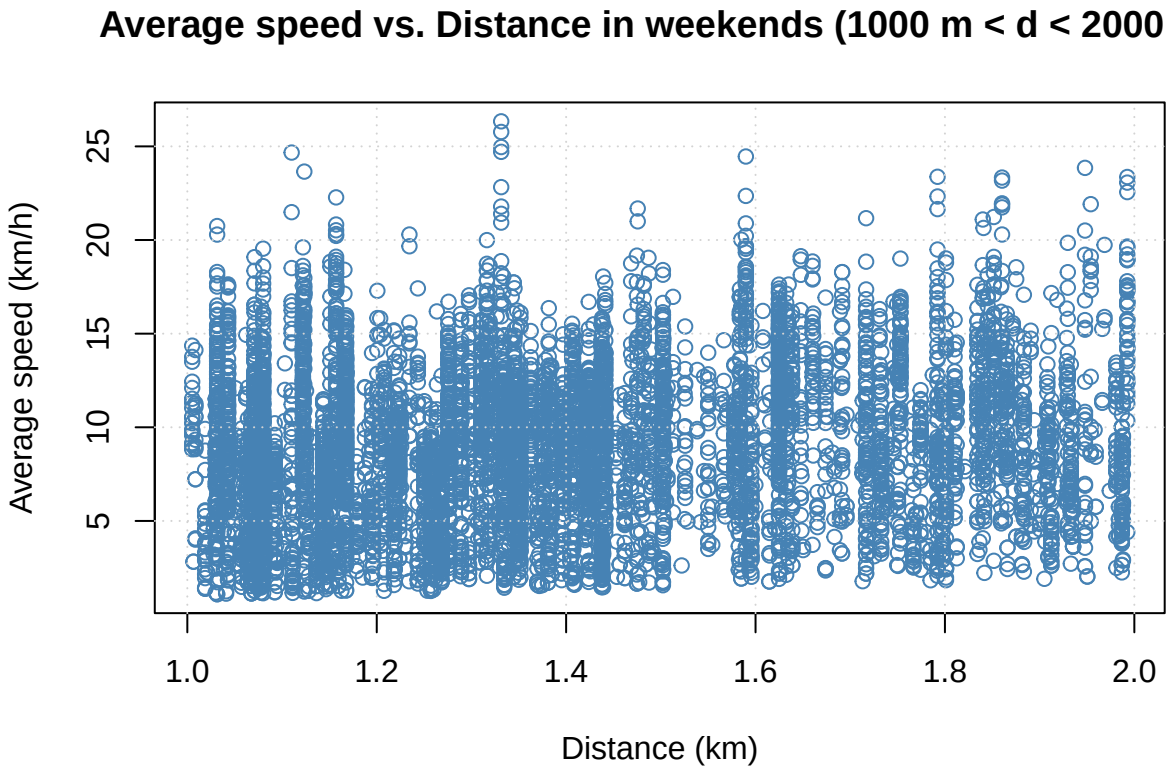

```
plot(distances[distances < 0.5 & !startmask],
     average_speed[distances < 0.5 & !startmask],
     main = "Average speed vs. Distance in weekends (d < 500 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



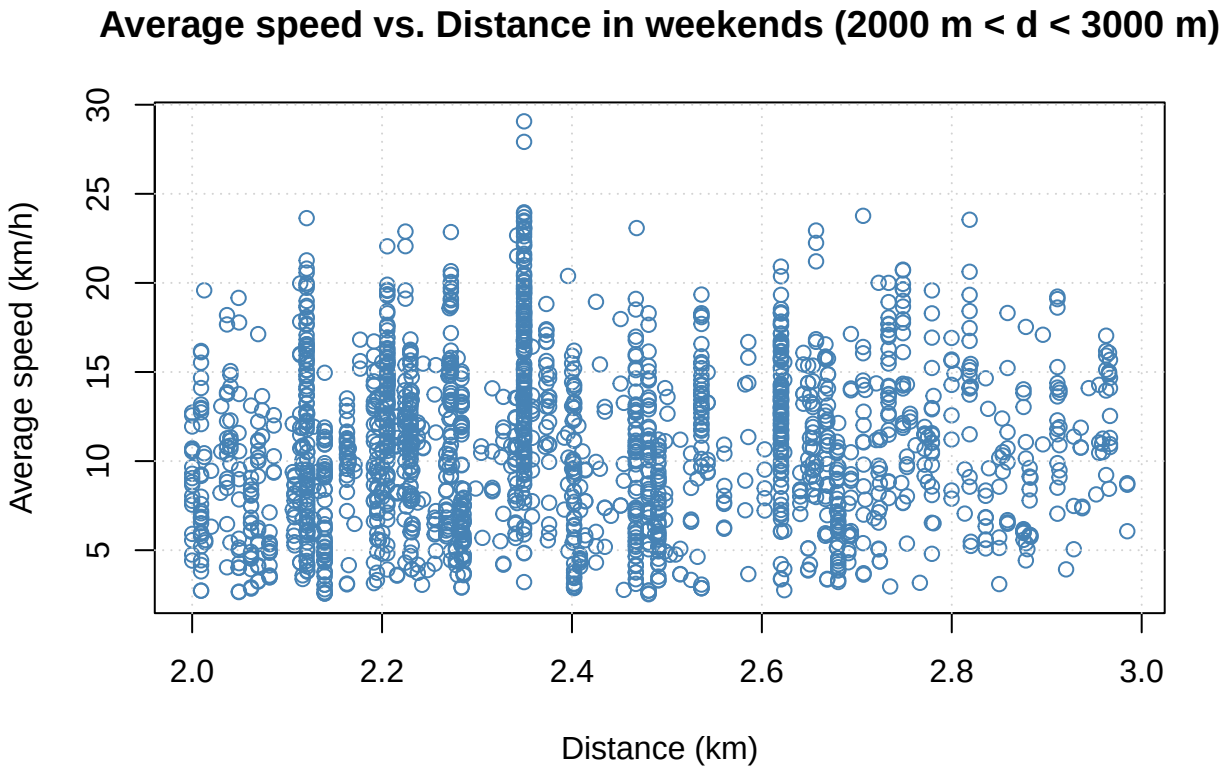

```
plot(distances[distances > 0.5 & distances < 1.0 & !startmask],
     average_speed[distances > 0.5 & distances < 1.0 & !startmask],
     main = "Average speed vs. Distance in weekends (500 m < d < 1000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



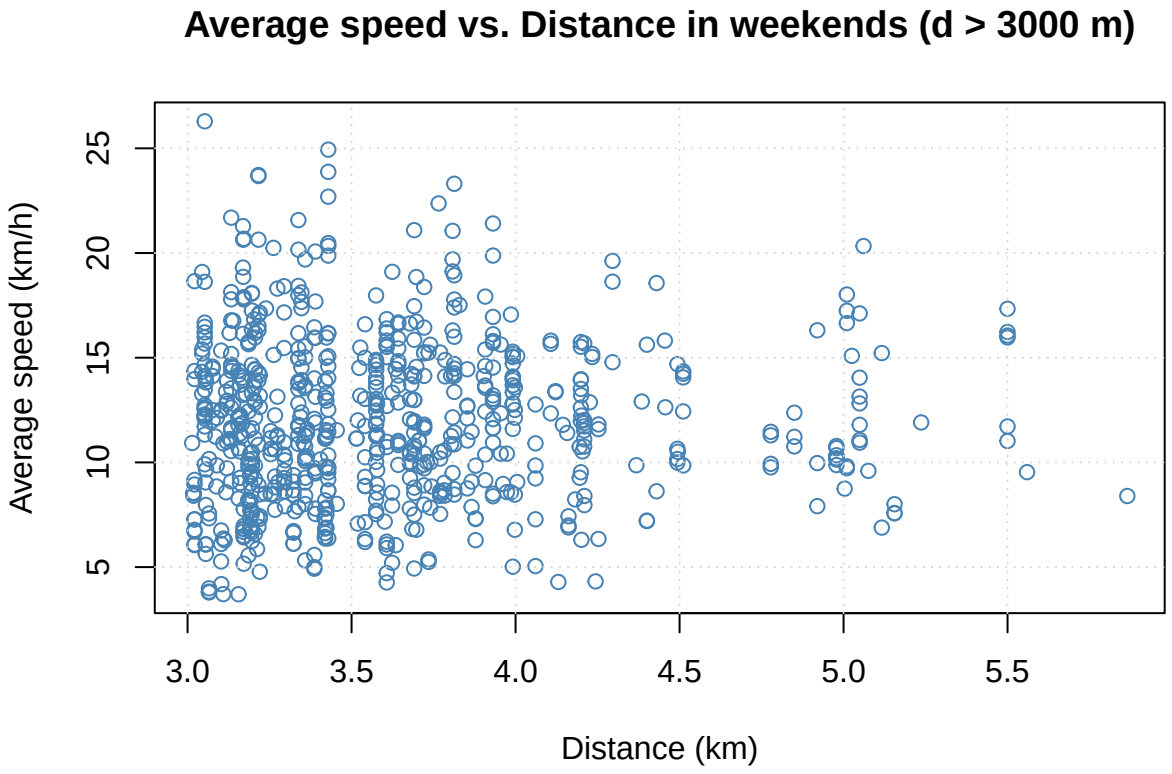

```r
plot(distances[distances > 1.0 & distances < 2.0 & !startmask],
     average_speed[distances > 1.0 & distances < 2.0 & !startmask],
     main = "Average speed vs. Distance in weekends (1000 m < d < 2000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```




```
plot(distances[distances > 2.0 & distances < 3.0 & !startmask],
     average_speed[distances > 2.0 & distances < 3.0 & !startmask],
     main = "Average speed vs. Distance in weekends (2000 m < d < 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```




```
plot(distances[distances > 3.0 & !startmask],
     average_speed[distances > 3.0 & !startmask],
     main = "Average speed vs. Distance in weekends (d > 3000 m)",
     xlab = "Distance (km)",
     ylab = "Average speed (km/h)",
     col  = "steelblue"
)
grid()
```



8.1 find the most common start station and the least popular end station

```
start_station_frequencies <- table(dataset$start.station.name)
end_station_frequencies   <- table(dataset$end.station.name)

print(sprintf("Most common start station: %s (%d times)", names(which.max(start_station_frequencies)), m
```

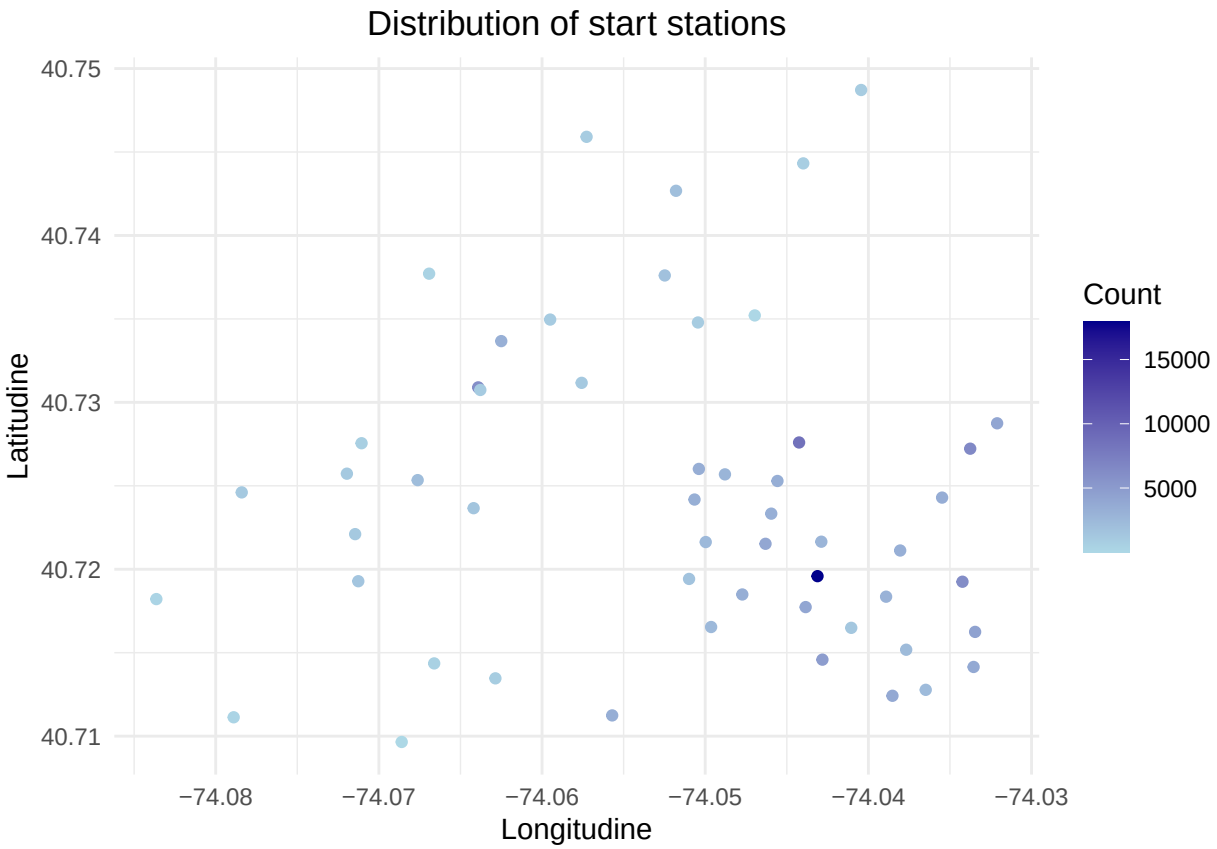
```
## [1] "Most common start station: Grove St PATH (17902 times)"
```

```
print(sprintf("Least popular end station: %s (%d time)", names(which.min(end_station_frequencies)), min
```

```
## [1] "Least popular end station: 1 Ave & E 16 St (1 time)"
```

8.2 show the distribution of start stations

```
start_station_count <- dataset[, c("start.station.longitude", "start.station.latitude")] %>%
                       group_by(start.station.longitude, start.station.latitude)         %>%
                       summarise(Count = n(), .groups = "drop")

ggplot(start_station_count, aes(x = start.station.longitude, y = start.station.latitude)) +
       geom_point(aes(color = Count)) +
       scale_color_gradient(low = "lightblue", high = "darkblue") +
       labs(title = "Distribution of start stations", x = "Longitudine", y = "Latitudine") +
       theme_minimal() +
       theme(plot.title = element_text(hjust = 0.5))
```



8.3 find the three most common routes (start and end station) and the three least popular ones

```
route_counter <- dataset %>%
                 count(start.station.name, end.station.name, name = "count") %>%
                 arrange(desc(count))

print("Three most common routes:")
```

```
## [1] "Three most common routes:"
```

```
for (i in 1:3) {
  print(sprintf("%s -> %s (%d times)", route_counter$start.station.name[i], route_counter$end.station.na
}
```

```
## [1] "Hamilton Park -> Grove St PATH (3037 times)"
## [1] "Grove St PATH -> Hamilton Park (2318 times)"
## [1] "Brunswick & 6th -> Grove St PATH (1916 times)"
```

```
cat("\n")
```

```
print("Three least popular routes:")
```

```
## [1] "Three least popular routes:"
```

```
for (i in 0:2) {
  j <- nrow(route_counter) - i
  print(sprintf("%s -> %s (%d time)", route_counter$start.station.name[j], route_counter$end.station.nar
```

```
}
```

```
## [1] "York St -> Lincoln Park (1 time)"
## [1] "York St -> Communipaw & Berry Lane (1 time)"
## [1] "York St -> Brunswick & 6th (1 time)"
```