Assignment 1

Exercise 1

The repository https://drive.google.com/drive/folders/1NESuaJ5yGIrAli1TgrpnK5hnoxGsMi3h?usp=sharing contains bike-sharing data provided by New York City, Citi Bike sharing system. The data (in csv format) is structured as follows:

- Trip duration (in seconds)
- Start Time and date
- Stop Time and date
- Start Station ID, name, latitude and longitude
- End Station ID, name, latitude and longitude
- Bike ID
- User Type (Customer or Subscriber)
- Birth's Year
- Gender (0=unknown; 1=male; 2=female)

Points 1-3

1. Read the data and import them in a data.frame or tibble structure;
2. Merge the five data frames in a unique structure;
3. Check for missing data and remove it.

The files are stored inside the **Data_CitiBike*> **folder, the named** JC-20190x-citibike-tripdata.csv** with x in {2, 3, 4, 5, 6} being the identifier of the file.

```
# ---------------- READING AND IMPORTING DATAFRAMES ----------------
PATH <- "./Data_CitiBike/"
indices <- 2:6
files <- unlist(lapply(indices, function(i) {
  sprintf("%sJC-20190%i-citibike-tripdata.csv", PATH, i)
}))

dataframes <- lapply(files, function(filename) {
  read.csv(filename, header = TRUE, sep = ",")
})

# Merging
df <- bind_rows(dataframes)

# Checking for missing data
#fill blanks with not a numbers
df[df==""]<-NA
#complete.cases() gives TRUE if the row contains all the fields
df<-df[complete.cases(df),]

head(df)
```

```
##   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
##   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
##   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
##   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
```

Point 4

4.1 Compute the average and the median trip duration in minutes

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

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

```
writeLines(sprintf("Average trip duration:   %.1f min
                  \nMedian trip duration:    %.1f min
                  \nMaximum trip duration:   %.1f min
                  \nMinimum trip duration:   %.1f min",
               mean(df$tripduration)/60.,
               median(df$tripduration)/60.,
               max(df$tripduration)/60.,
               min(df$tripduration)/60.))
```

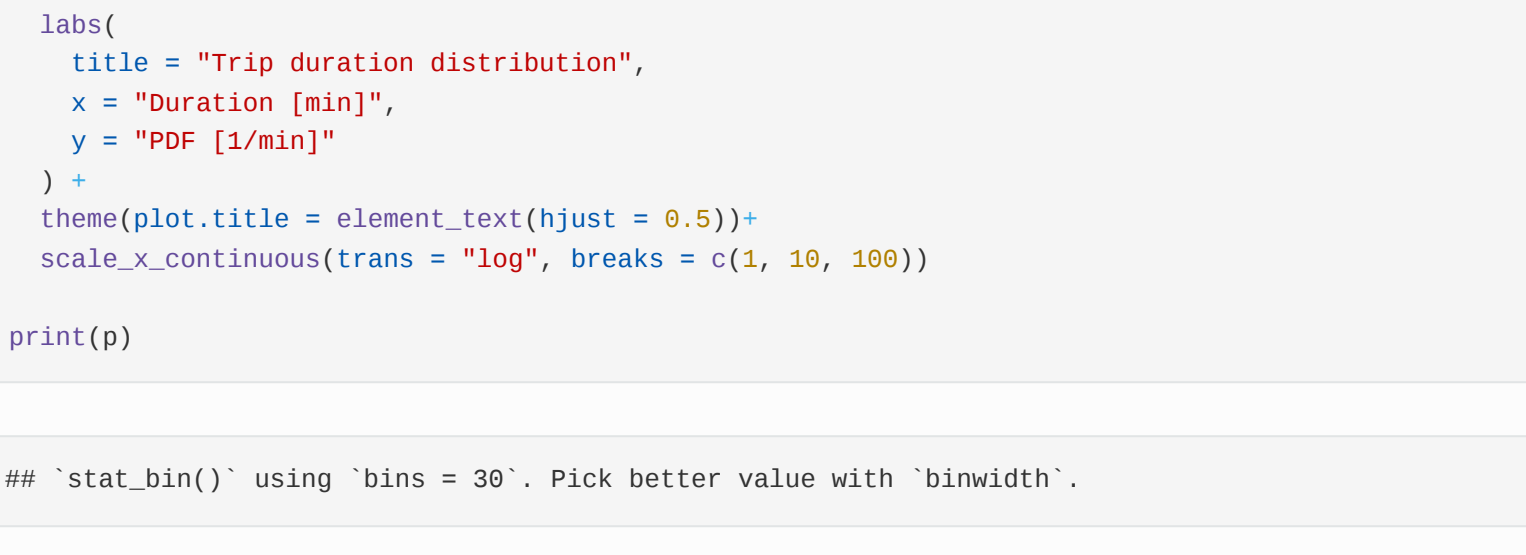
```
## Average trip duration:   12.8 min
##
## Median trip duration:    5.7 min
##
## Maximum trip duration:   28817.0 min
##
## Minimum trip duration:   1.0 min
```

```
skimmed_df = df[df$tripduration<(3.*3600), ]
writeLines(sprintf(
             "Average trip duration:   %.1f min
            \nMedian trip duration:    %.1f min
            \nMaximum trip duration:   %.1f min
            \nMinimum trip duration:   %.1f min
            \nNumber of rejected rows: %i",
              mean(skimmed_df$tripduration)/60.,
              median(skimmed_df$tripduration)/60.,
              max(skimmed_df$tripduration)/60.,
              min(skimmed_df$tripduration)/60.,
              nrow(df) - nrow(skimmed_df)))
```

```
## Average trip duration:   9.2 min
##
## Median trip duration:    5.7 min
##
## Maximum trip duration:   180.0 min
##
## Minimum trip duration:   1.0 min
##
## Number of rejected rows: 428
```

```
p<-ggplot(skimmed_df, aes(tripduration/60, y = after_stat(density))) +
  geom_histogram(color="black", fill=color_vector[6]) +
  geom_density(alpha=.3, fill=color_vector[2], linetype="dashed") +
  labs(
    title = "Trip duration distribution",
    x = "Duration [min]",
    y = "PDF [1/min]"
  ) +
  theme(plot.title = element_text(hjust = 0.5))+
  scale_x_continuous(trans = "log", breaks = c(1, 10, 100))

print(p)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

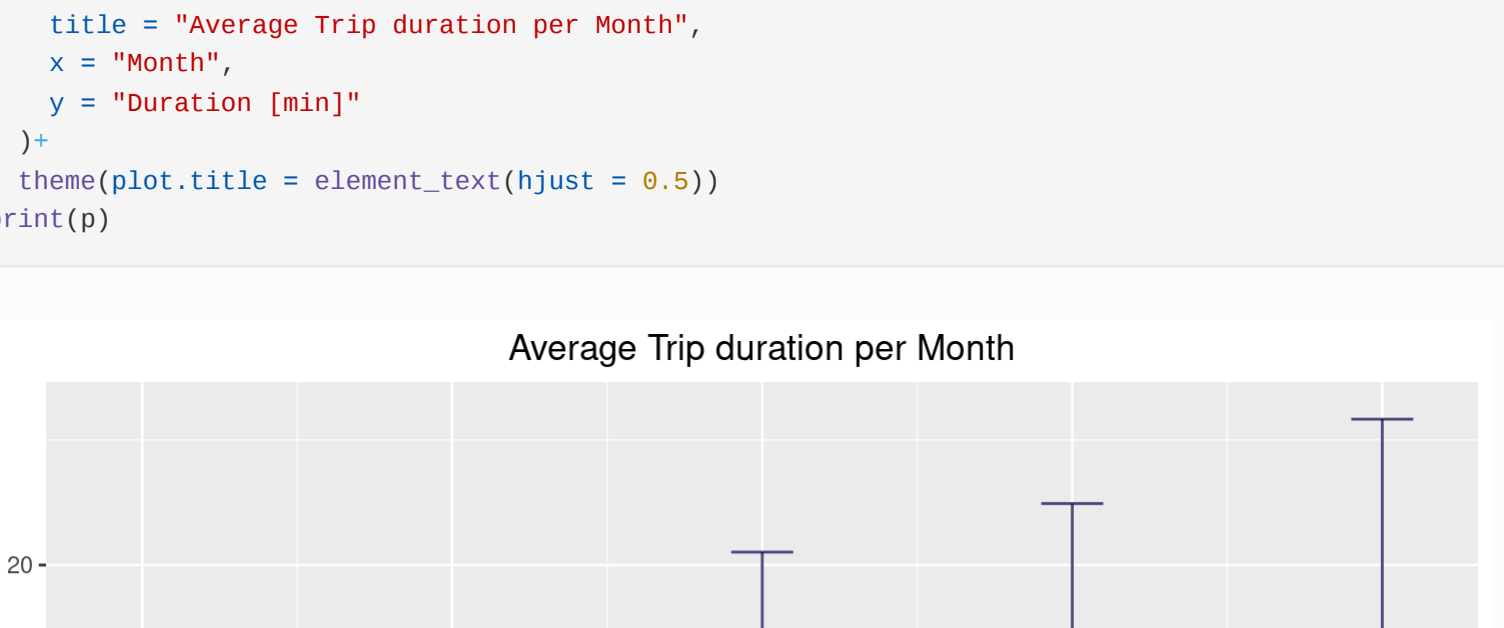


Point 5

Plot the monthly average trip duration

```
month <- unlist(lapply(skimmed_df$starttime, function(date) {
  as.POSIXlt(date)$mon
}))

skimmed_df$month = month
```

```
monthly_trip <- skimmed_df %>%
  group_by(month) %>%
  summarise(average_duration = mean(tripduration) / 60, std = sd(tripduration)/60)

p<- ggplot(monthly_trip, aes(x = month, y = average_duration)) +
  geom_line(color=color_vector[1]) +
  geom_point(color=color_vector[7])+
  geom_errorbar(aes(ymin=average_duration-std, ymax=average_duration+std), width=.2, color = color_vec
  position=position_dodge(0.05)) +
  labs(
    title = "Average Trip duration per Month",
    x = "Month",
    y = "Duration [min]"
  )+
  theme(plot.title = element_text(hjust = 0.5))
print(p)
```



Point 6

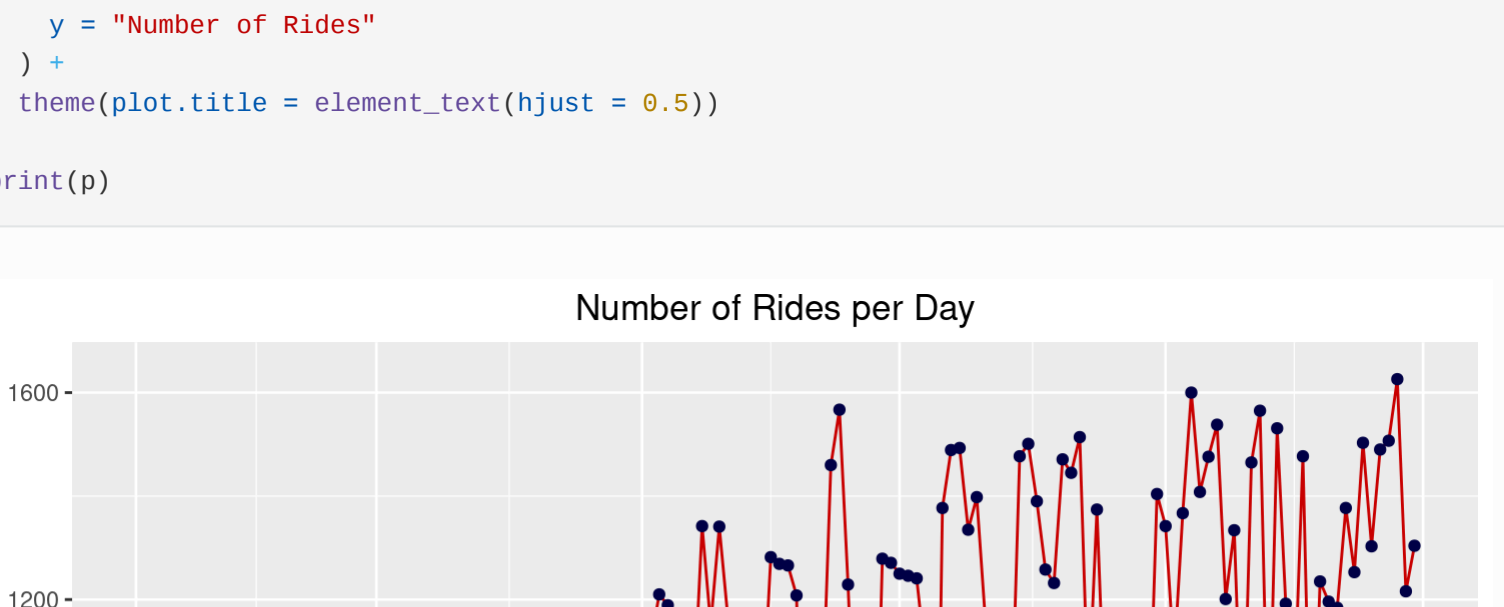
6.1 Plot the number of rides per day

6.2 Plot the hourly distribution on weekdays and weekends

6.3 Plot the hourly distribution on weekdays and weekends separating by customers and users

```
date <- unlist(lapply(skimmed_df$starttime, function(date) {
  as.POSIXct(date, format = "%Y-%m-%d")
}))

skimmed_df$date = date

hour <- as.numeric(format(as.POSIXct(skimmed_df$starttime), format = "%H"))
skimmed_df$hour <- hour

day_of_week <- weekdays(as.POSIXct(skimmed_df$starttime))
skimmed_df$day_of_week <- day_of_week

weekends_list = c('sabato', 'domenica')
weekends <- skimmed_df$day_of_week %in% weekends_list
skimmed_df$weekends = weekends
```

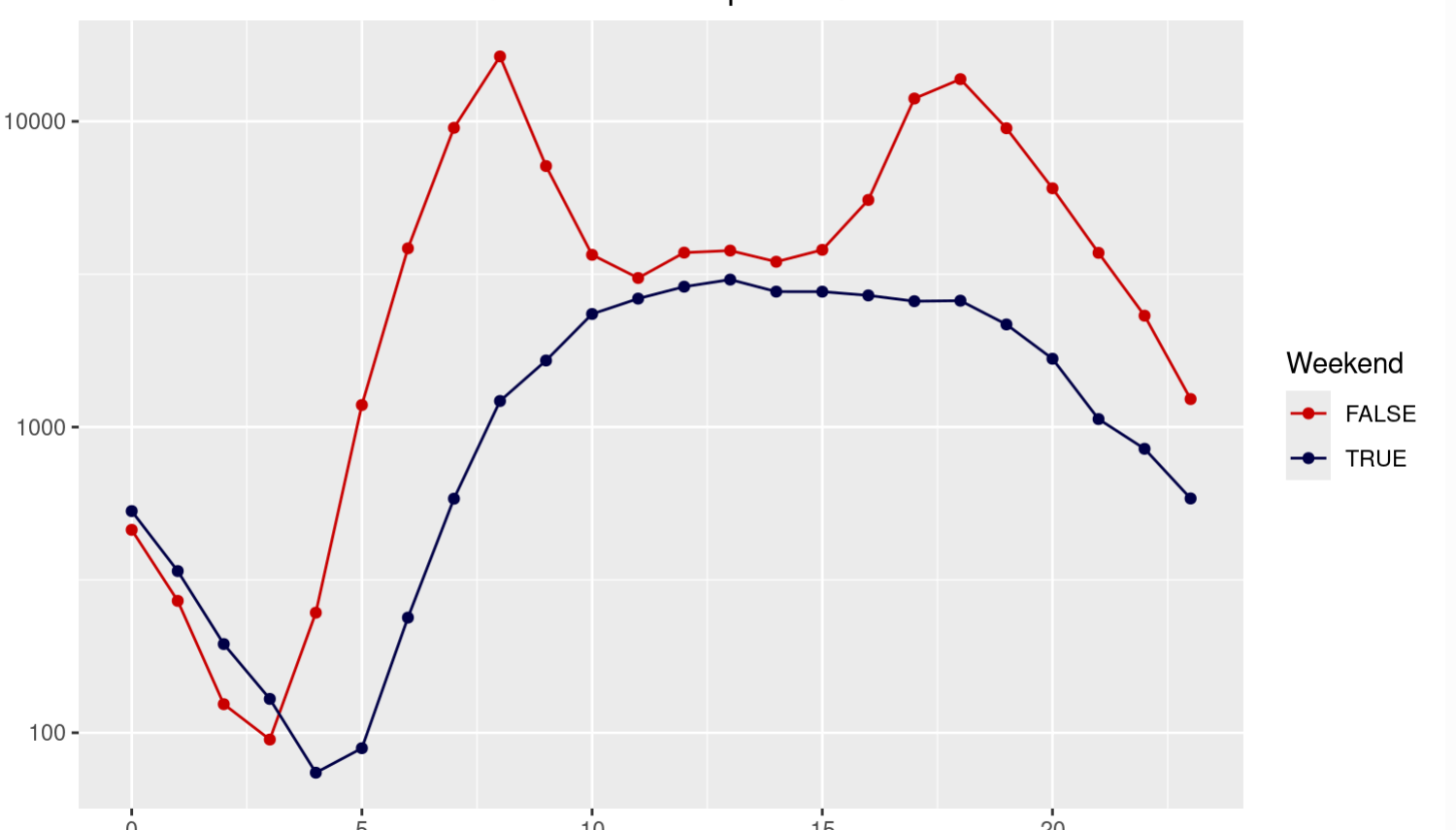
```
rides_per_day <- skimmed_df %>%
                group_by(date) %>%
                summarise(rides = n())

p <- ggplot(rides_per_day, aes(x = as.POSIXct(date), y = rides)) +
  geom_line(color = color_vector[1])  +
  geom_point(color = color_vector[7]) +
  labs(
    title = "Number of Rides per Day",
    x = "Date",
    y = "Number of Rides"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

print(p)
```




```
hourly_distro <- skimmed_df %>%
                group_by(weekends, hour) %>%
                summarise(rides = n())
```

```
## `summarise()` has grouped output by 'weekends'. You can override using the
## `.groups` argument.
```

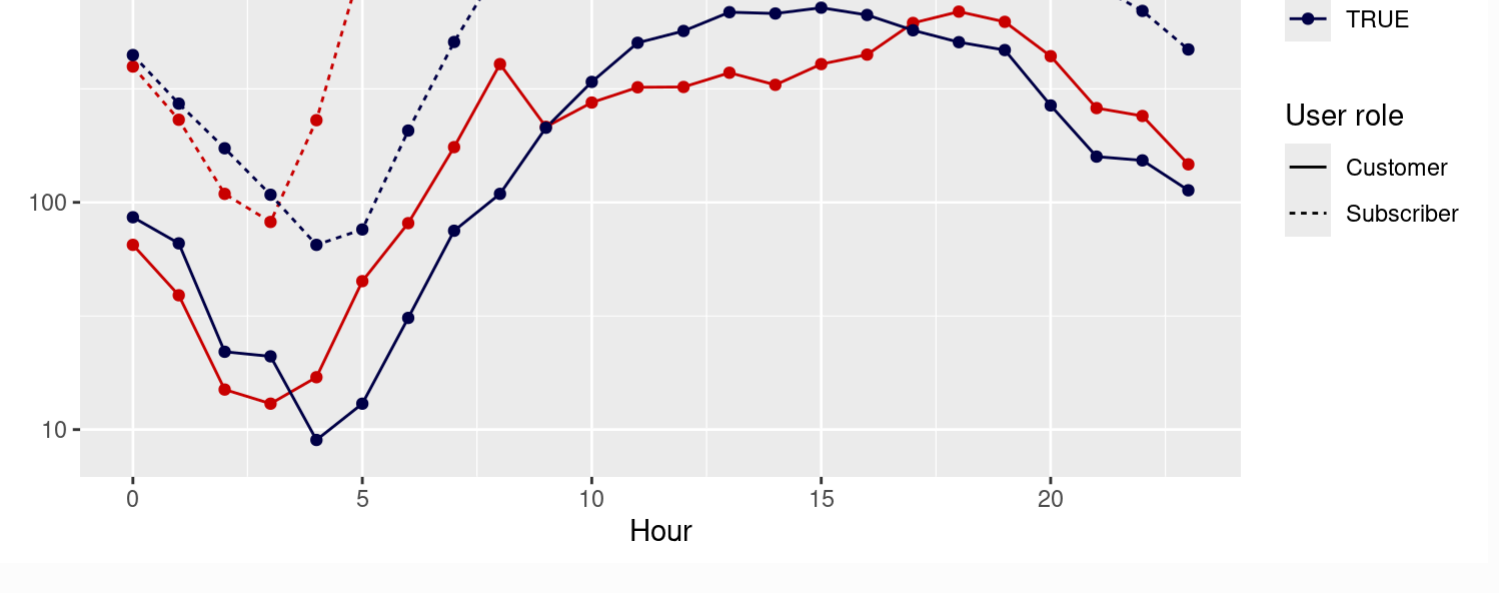
```
p <- ggplot(hourly_distro, aes(x = hour, y = rides, color = factor(weekends))) +
  geom_line()  +
  geom_point(aes(color = factor(weekends))) +

  labs(
    title = "Number of Rides per Hour",
    x = "Hour",
    y = "Number of Rides",
    color = "Weekend"
  ) +
  scale_color_manual(values = c(color_vector[1], color_vector[7])) +
  scale_y_continuous(trans = "log", breaks = c(10, 100, 1000, 10000, 100000)) +

  theme(plot.title = element_text(hjust = 0.5))

print(p)
```




```
hourly_distro <- skimmed_df %>%
                group_by(weekends, hour, usertype) %>%
                summarise(rides = n())
```

```
## `summarise()` has grouped output by 'weekends', 'hour'. You can override using
## the `.groups` argument.
```

```
p <- ggplot(hourly_distro, aes(x = hour, y = rides, color = factor(weekends), linetype = factor(userty
  geom_line()  +
  geom_point(aes(color = factor(weekends))) +

  labs(
    title = "Number of Rides per Hour",
    x = "Hour",
    y = "Number of Rides",
    color = "Weekend",
    linetype = "User role"
  ) +
  scale_color_manual(values = c(color_vector[1], color_vector[7])) +
  scale_y_continuous(trans = "log", breaks = c(10, 100, 1000, 10000, 100000)) +
  theme(plot.title = element_text(hjust = 0.5))

print(p)
```



Point 7

7.1 Evaluate the average speed of users

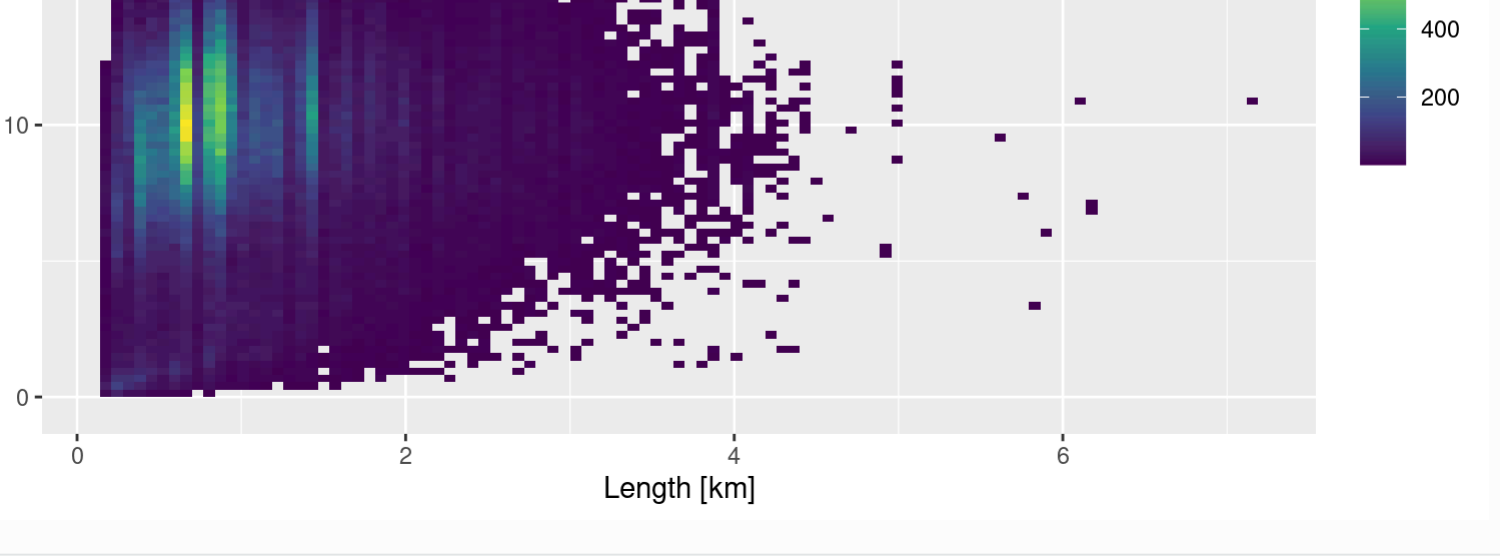
7.2 Plot the average speed as a function of route length

7.3 Separate the results for weekdays and weekends

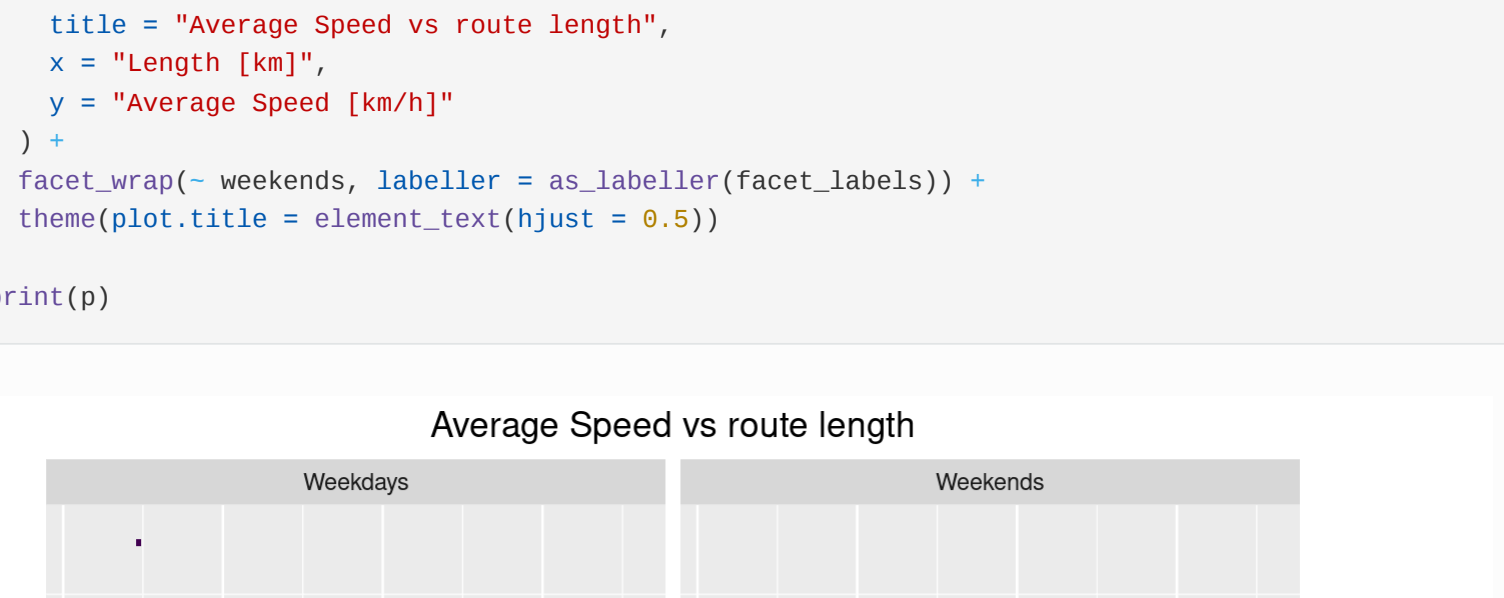
```
distances <- unlist(
  mapply(
    function(lon1, lat1, lon2, lat2){
      distHaversine(c(lon1, lat1), c(lon2, lat2))
    },
    skimmed_df$start.station.longitude, skimmed_df$start.station.latitude, skimmed_df$end.station.long
  ))
skimmed_df$distances <- distances
average_speed <- skimmed_df$distances / skimmed_df$tripduration
skimmed_df$average_speed = average_speed
writeLines(sprintf(
             "Average speed:   %.1f m/s",
             mean(average_speed)
             ))
```

```
## Average speed:   2.5 m/s
```

```
p <- ggplot(skimmed_df[skimmed_df$average_speed*3.6<30 & skimmed_df$average_speed>0 & skimmed_df$dista
  geom_bin2d(bins = 100) +
  scale_fill_continuous(type = "viridis") +
  labs(
    title = "Average Speed vs route length",
    x = "Length [km]",
    y = "Average Speed [km/h]"
  ) +
  theme(plot.title = element_text(hjust = 0.5))

print(p)
```




```
facet_labels <- c(
  "TRUE"  = "Weekends",
  "FALSE" = "Weekdays"
)
p <- ggplot(skimmed_df[skimmed_df$average_speed*3.6<30 & skimmed_df$average_speed>0 & skimmed_df$dista
  geom_bin2d(bins = 100, aes(fill = after_stat(density))) +
  scale_fill_continuous(type = "viridis", name="Density") +
  labs(
    title = "Average Speed vs route length",
    x = "Length [km]",
    y = "Average Speed [km/h]"
  ) +
  facet_wrap(~ weekends, labeller = as_labeller(facet_labels)) +
  theme(plot.title = element_text(hjust = 0.5))

print(p)
```



Point 8

8.1 Find the most common start station and the least popular end station.

8.2 Show the distribution of start stations

8.3 Find the three most common routes and the three least popular ones

```
start_station_count <- skimmed_df %>%
                       group_by(start.station.id) %>%
                       summarise(start_count = n()) %>%
                       arrange(desc(start_count))

end_station_count   <- skimmed_df %>%
                       group_by(end.station.id) %>%
                       summarise(end_count = n()) %>%
                       arrange(end_count)

route_count         <- skimmed_df %>%
                       group_by(start.station.id, end.station.id) %>%
                       summarise(route_count = n()) %>%
                       arrange(desc(route_count))
```

```
## `summarise()` has grouped output by 'start.station.id'. You can override using
## the `.groups` argument.
```

```
n_routes = nrow(route_count)
writeLines(
  sprintf(
"
The most common start station has ID: %i
The least popular end station has ID: %i
The three most common routes are (start_id, end_id):
  - (%i,%i);
  - (%i,%i);
  - (%i,%i).
The three least common routes are (start_id, end_id):
  - (%i,%i);
  - (%i,%i);
  - (%i,%i).
"   ,
    start_station_count[[1,1]],
    end_station_count[[1, 1]],

    route_count[[1, 1]], route_count[[1, 2]],
    route_count[[2, 1]], route_count[[2, 2]],
    route_count[[3, 1]], route_count[[3, 2]],

    route_count[[n_routes, 1]],   route_count[[n_routes, 2]],
    route_count[[n_routes-1, 1]], route_count[[n_routes-1, 2]],
    route_count[[n_routes-2, 1]], route_count[[n_routes-2, 2]]
  )
)
```

```
##
## The most common start station has ID: 3186
## The least popular end station has ID: 224
## The three most common routes are (start_id, end_id):
##   - (3203,3186);
##   - (3186,3203);
##   - (3269,3186).
## The three least common routes are (start_id, end_id):
##   - (3791,3679);
##   - (3791,3640);
##   - (3791,3483).
```

```
p<-ggplot(start_station_count, aes(x = as.character(start.station.id), y = start_count)) +
  geom_bar(stat = "identity", fill = color_vector[6], color = "black") +
  labs(
    title = "Distribution of start stations",
    x = "Start station ID",
    y = "Counts"
  ) +
  theme(plot.title = element_text(hjust = 0.5),
  axis.text.x = element_text(angle = 60, hjust = 1))

print(p)
```