STAT 33B Homework 3

Gunnar Mayer (3034535154)

This assignment is due **March 4, 2020** by 11:59pm.

The purpose of this assignment is to practice working with data frames, including loading tabular data, taking subsets, and making plots.

Edit this file, knit to PDF, and:

- Submit the Rmd file on bCourses.
- Submit the PDF file on Gradescope.

If you think you'll need help with submission, please ask in office hours *before* the assignment is due.

Answer all questions with complete sentences, and put code in code chunks. You can make as many new code chunks as you like. Please do not delete the exercises already in this notebook, because it may interfere with our grading tools.

The D.C. Bikeshare Data Set

In this assignment, you'll use data collected from the Capital Bikeshare System in Washington, D.C. in 2011 and 2012. The data set is split across two separate files: `hour.csv`, which contains hourly rider counts, and `day.csv`, which is aggregated to the daily level.

Both `hour.csv` and `day.csv` have the following fields:

- `instant`: record index
- `dteday`: date
- `season`: season (1: Winter, 2: Spring, 3: Summer, 4: Fall)
- `yr`: year (0: 2011, 1: 2012)
- `mnth`: month (1 to 12)
- `holiday`: 1 if day is holiday, 0 otherwise
- `weekday`: day of the week (0: Sunday, 1: Monday, . . .)
- `workingday`: 1 if day is neither weekend nor holiday, 0 otherwise
- `weathersit` :
  - 1: Clear or Partly Cloudy
  - 2: Cloudy or Misty
  - 3: Light Rain or Snow (including Thunderstorms)
  - 4: Heavy Rain, Hail, or Snow (including Thunderstorms)
- `temp`: Normalized temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min), t_min=-8, t_max=+39 (only in hourly scale)
- `atemp`: Normalized feeling temperature in Celsius. The values are derived via (t-t_min)/(t_max-t_min), t_min=-16, t_max=+50 (only in hourly scale)
- `hum`: Normalized humidity. The values are divided to 100 (max)
- `windspeed`: Normalized wind speed. The values are divided to 67 (max)

- `casual`: count of casual users
- `registered`: count of registered users (regular subscribers)
- `cnt`: count of total rental bikes including both casual and registered

The `hour.csv` additionally has the field:

- `hr` : hour (0 to 23)

Details about the sources for this data set are available here.

Exercise 1

The comma-separated values (CSV) format does not include information about the data type of each column. When you load a CSV file into R, R tries to guess the correct data type for each column.

R usually does a pretty good job, but sometimes it's necessary to manually convert some columns to the correct data type.

For this exercise:

1. Read the `day.csv` file into R with `read.csv()`. Assign it to the variable `bike_day`.
2. The `holiday` and `workingday` columns are recorded in the CSV file as integers, but represent logical data. Use the `as.logical()` function to convert these columns to logical data (replacing the original columns).
3. The `season`, `weekday`, and `weathersit` columns are recorded in the CSV file as integers, but represent categorical data. Use the `factor()` function to convert these columns to factors with appropriately-labelled levels.
4. R guesses that the `dteday` column is categorical data (a factor) even though it is actually a date. Use the `as.Date()` function to convert the column to the `Date` class.
5. The `year` column is recorded in the CSV with integers 0 and 1 to represent the years 2011 and 2012, respectively. Use subset assignment to change the values equal to 0 to 2011, and the values equal to 1 to 2012 (keep the column as integer type).

```
# 1
bike_day = read.csv("day.csv")

#2
bike_day$holiday = as.logical(bike_day$holiday)
bike_day$workingday = as.logical(bike_day$workingday)

#3
bike_day$season = factor(bike_day$season, levels = c(1, 2, 3, 4),
   labels = c("Winter", "Spring", "Summer", "Fall"))
bike_day$weekday = factor(bike_day$weekday, levels = c(0, 1, 2, 3, 4, 5, 6),
   labels = c("Sunday","Monday", "Tuesday", "Wednesday", "Thursday",
              "Friday", "Saturday"))
bike_day$weathersit = factor(bike_day$weathersit, levels = c(1,2,3,4),
   labels = c("Clear or Partly Cloudy", "Cloudy or Misty",
   "Light Rain or Snow (including Thunderstorms)",
   "Heavy Rain, Hail, or Snow (including Thunderstorms)"))
```

```r
# 4
bike_day$dteday = as.Date(bike_day$dteday)

#5
bike_day$yr[bike_day$yr == 0] = 2011L
bike_day$yr[bike_day$yr == 0] = 2012L
```

Exercise 2

Repeat Steps 1-5 in Exercise 1 for the `hour.csv` file. Assign the resulting data frame to a variable named `bike`.

```r
# 1
bike = read.csv("hour.csv")

#2
bike$holiday = as.logical(bike$holiday)
bike$workingday = as.logical(bike$workingday)

#3
bike$season = factor(bike$season, levels = c(1, 2, 3, 4),
   labels = c("Winter", "Spring", "Summer", "Fall"))
bike$weekday = factor(bike$weekday, levels = c(0, 1, 2, 3, 4, 5, 6),
   labels = c("Sunday", "Monday", "Tuesday", "Wednesday",
              "Thursday", "Friday", "Saturday"))
bike$weathersit = factor(bike$weathersit, levels = c(1,2,3,4),
   labels = c("Clear or Partly Cloudy", "Cloudy or Misty",
   "Light Rain or Snow (including Thunderstorms)",
   "Heavy Rain, Hail, or Snow (including Thunderstorms)"))

# 4
bike$dteday = as.Date(bike$dteday)

#5
bike$yr[bike_day$yr == 0] = 2011L
bike$yr[bike_day$yr == 0] = 2012L
```

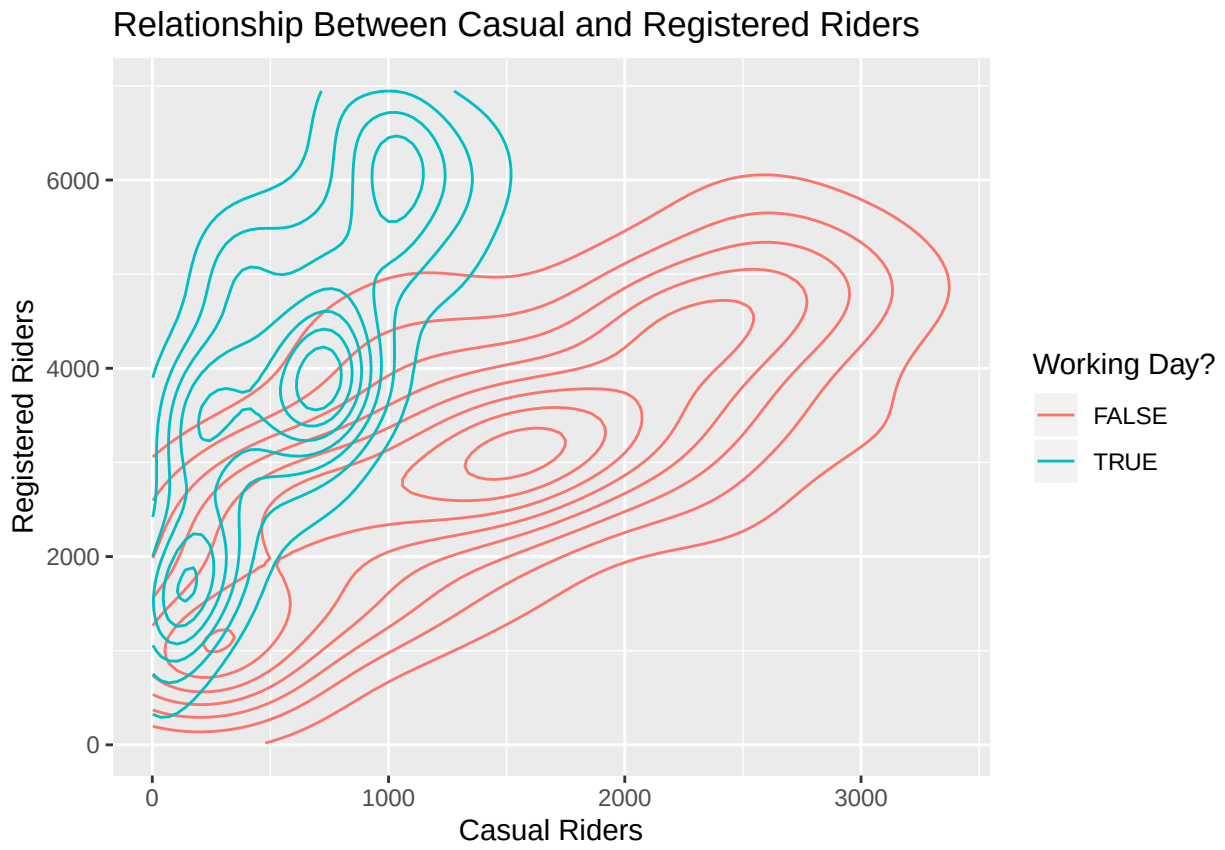
Exercise 3

Investigate the relationship between casual and registered riders in the `bike_day` data frame. Make a 2-dimensional kernel density plot of `casual` versus `registered` with `geom_density_2d()`. This kind of plot uses contour lines to represent the concentration of points, similar to how a topographic map uses contour lines to represent height. Use color to distinguish between the rides that took place on working days and non-working days. Add a title and axis labels to your plot.

```r
library(ggplot2)

gd = ggplot(bike_day, aes(x = casual, y = registered)) +
      geom_density_2d(aes(color=workingday)) +
      labs(title = "Relationship Between Casual and Registered Riders") +
```

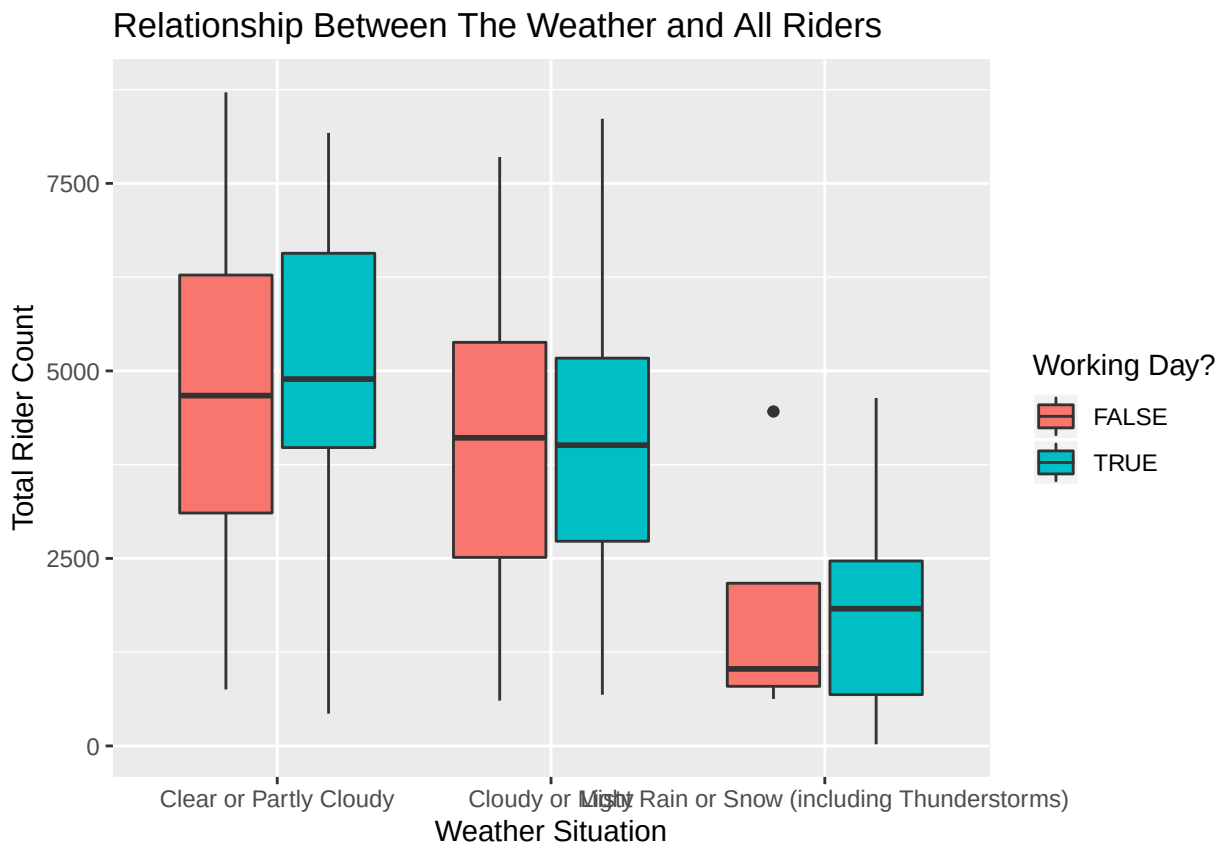
```
      labs(x = "Casual Riders", y = "Registered Riders", color = "Working Day?")
gd
```



Exercise 4

Investigate the relationship between weather and riders in the `bike_day` data frame. Make side-by-side box plots with `geom_boxplot()` that show the distribution of total riders for each weather situation. Use fill color to distinguish between working days and non-working days. Add a title and axis labels to your plot.

```
gb = ggplot(bike_day, aes(x = weathersit, y = cnt, fill = workingday)) +
      geom_boxplot() +
      labs(title = "Relationship Between The Weather and All Riders") +
      labs(x = "Weather Situation", y = "Total Rider Count",  fill = "Working Day?")
gb
```



Exercise 5

Occasionally, it's necessary to reshape a data frame in order to create a particular plot.

Reshape the `bike` data frame so that the registered and casual rider counts are in the same column, called `count`. The reshaped data frame should have another new column called `rider` that is either "registered" or "casual", depending on which of the original columns the `count` value came from. Assign the reshaped data frame to the variable `bike_stack`.

The `hr`, `weekday`, `rider`, and `count` columns of `bike_stack` should be similar to this:

```
  hr  weekday      rider count
1  0 Saturday     casual     3
2  0 Saturday registered    13
3  1 Saturday     casual     8
4  1 Saturday registered    32
5  2 Saturday     casual     5
6  2 Saturday registered    27
```

With the `bike_stack` data, use `geom_smooth()` to plot a smooth curves fitted to the counts over the hour of the day, with separate lines for casual and registered riders. Add a title and axis labels to your plot.

```
library(tidyr)
bike_stack = pivot_longer(bike, cols = casual:registered,
      names_to = "Riders", values_to = "Count")
```

```
gs = ggplot(bike_stack, aes(x = hr, y = Count)) +
   geom_smooth() +
   labs(title = "Number of Riders vs Time of the Day",
   x = "Hour of the Day", y = "Count of Riders" )
gs
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```