R project

Qianqian Meng

TASK 1: Manipulation

Q1. Load the dataset dublin-bikes.txt, save it as a tibble and give meaningful names to the variables related to the weather.

```
library(rio)
dublin_bikes <- import("dublin-bikes-v2.txt", setclass = "tibble") #import as tibble

dublin_bikes <- dublin_bikes |>
  dplyr::rename(rainfall = rain,
         air_temp = temp,
         wind_speed = wdsp,
         cloud_amount = clamt
         ) #rename weather variables
names(dublin_bikes) #check my variables
```

```
 [1] "Time"                           "rainfall"
 [3] "air_temp"                       "wind_speed"
 [5] "cloud_amount"                   "Clontarf - James Larkin Rd"
 [7] "Clontarf - Pebble Beach Carpark" "Griffith Avenue (Clare Rd Side)"
 [9] "Griffith Avenue (Lane Side)"    "Grove Road Totem"
[11] "Richmond Street Cyclists 1"     "Richmond Street Cyclists 2"
```

Q2. What is the size (number of rows and columns) this dataset? Write some code to check that the variable Time is stored using an appropriate class for a date, and the other variables are numeric, fix them if they aren't.

size: 8760 rows and 12 columns

Time is stored using class POSIXct, other variables are all numeric

```
str(dublin_bikes)
```

```
tibble [8,760 x 12] (S3: tbl_df/tbl/data.frame)
 $ Time                           : POSIXct[1:8760], format: "2022-09-01 00:00:00" "2022-09-0
 $ rainfall                       : num [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ air_temp                       : num [1:8760] 13 13.6 14 14.4 14.4 13.5 14.2 15 15.5 16.4
 $ wind_speed                     : int [1:8760] 6 7 6 5 5 6 6 8 9 11 ...
 $ cloud_amount                   : int [1:8760] 6 6 6 6 7 7 7 7 7 5 ...
 $ Clontarf - James Larkin Rd     : int [1:8760] 6 1 1 0 1 21 30 89 123 67 ...
 $ Clontarf - Pebble Beach Carpark: int [1:8760] 8 2 2 0 3 26 44 111 170 90 ...
 $ Griffith Avenue (Clare Rd Side): int [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ Griffith Avenue (Lane Side)    : int [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ Grove Road Totem               : int [1:8760] 33 8 5 6 2 39 132 324 619 287 ...
 $ Richmond Street Cyclists 1     : int [1:8760] 25 3 7 7 2 2 9 43 81 42 ...
 $ Richmond Street Cyclists 2     : int [1:8760] 8 1 6 3 2 9 40 88 228 153 ...
```

Q3. Convert the variable containing the cloud amount information into an ordered factor. Print the levels and the output of a check to confirm it's ordered.

```
dublin_bikes$cloud_amount <- factor(dublin_bikes$cloud_amount, ordered = TRUE)
levels(dublin_bikes$cloud_amount)
```

```
[1] "0" "1" "2" "3" "4" "5" "6" "7" "8"
```

```
str(dublin_bikes$cloud_amount) # Confirm
```

```
 Ord.factor w/ 9 levels "0"<"1"<"2"<"3"<..: 7 7 7 7 8 8 8 8 8 6 ...
```

Q4. Split the information in the column Time into two columns: one containing the date (i.e. date only, no time), and the other the hour. Check that there are 24 hours for each date, and that there are 365 different dates.

```
library(lubridate)
```

```
Attaching package: 'lubridate'

The following objects are masked from 'package:base':

    date, intersect, setdiff, union
```

```
dublin_bikes$date <- date(dublin_bikes$Time)
dublin_bikes$hour<- hour(dublin_bikes$Time)
tab1 <- table(dublin_bikes$date) #frequncy table
str(tab1) #there are 24 hours for each date, since the data recorded hourly
```

```
 'table' int [1:365(1d)] 24 24 24 24 24 24 24 24 24 24 ...
 - attr(*, "dimnames")=List of 1
  ..$ : chr [1:365] "2022-09-01" "2022-09-02" "2022-09-03" "2022-09-04" ...
```

```
length(tab1) #there are 365 different dates
```

```
[1] 365
```

Q5. Add two columns one containing the day of the week and the other the month. Check that these two columns are ordered factors.

```
dublin_bikes$day_of_week = wday(dublin_bikes$date, label=TRUE, week_start = 1)#sets Monday
dublin_bikes$month = month(dublin_bikes$date, label=TRUE) #return month names instead of n

str(dublin_bikes$day_of_week)
```

```
 Ord.factor w/ 7 levels "Mon"<"Tue"<"Wed"<..: 4 4 4 4 4 4 4 4 4 4 ...
```

```
str(dublin_bikes$month)
```

```
 Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 9 9 9 9 9 9 9 9 9 9 ...
```

Q6. Remove the column Time and use dplyr::relocate() to put the new columns with the date, hour, day of the week, and month as the first four columns of the dataset.

```
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
```

```r
dublin_bikes <- dublin_bikes |>
  select(-Time) |> #not include Time column
  dplyr::relocate('date', 'hour', 'day_of_week', 'month', .before=1) #first few columns .b

str(dublin_bikes)
```

```
tibble [8,760 x 15] (S3: tbl_df/tbl/data.frame)
 $ date                          : Date[1:8760], format: "2022-09-01" "2022-09-01" ...
 $ hour                          : int [1:8760] 0 1 2 3 4 5 6 7 8 9 ...
 $ day_of_week                   : Ord.factor w/ 7 levels "Mon"<"Tue"<"Wed"<..: 4 4 4 4 4 4
 $ month                         : Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 9 9 9 9 9 9
 $ rainfall                      : num [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ air_temp                      : num [1:8760] 13 13.6 14 14.4 14.4 13.5 14.2 15 15.5 16.4
 $ wind_speed                    : int [1:8760] 6 7 6 5 5 6 6 8 9 11 ...
 $ cloud_amount                  : Ord.factor w/ 9 levels "0"<"1"<"2"<"3"<..: 7 7 7 7 8 8 8
 $ Clontarf - James Larkin Rd    : int [1:8760] 6 1 1 0 1 21 30 89 123 67 ...
 $ Clontarf - Pebble Beach Carpark: int [1:8760] 8 2 2 0 3 26 44 111 170 90 ...
 $ Griffith Avenue (Clare Rd Side): int [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ Griffith Avenue (Lane Side)   : int [1:8760] 0 0 0 0 0 0 0 0 0 0 ...
 $ Grove Road Totem              : int [1:8760] 33 8 5 6 2 39 132 324 619 287 ...
 $ Richmond Street Cyclists 1    : int [1:8760] 25 3 7 7 2 2 9 43 81 42 ...
 $ Richmond Street Cyclists 2    : int [1:8760] 8 1 6 3 2 9 40 88 228 153 ...
```

Task 2: Analysis

Q1. Use functions from base R to compute which month had in total the highest and the lowest Precipitation Amount.

July had in total the highest Precipitation Amount.

February had in total the lowest Precipitation Amount.

```r
total_rainfall <- aggregate(dublin_bikes$rainfall,list(dublin_bikes$month),sum) #group by
higest = total_rainfall$Group.1[total_rainfall$x== max(total_rainfall$x)]
lowest = total_rainfall$Group.1[total_rainfall$x== min(total_rainfall$x)]

print(higest)
```

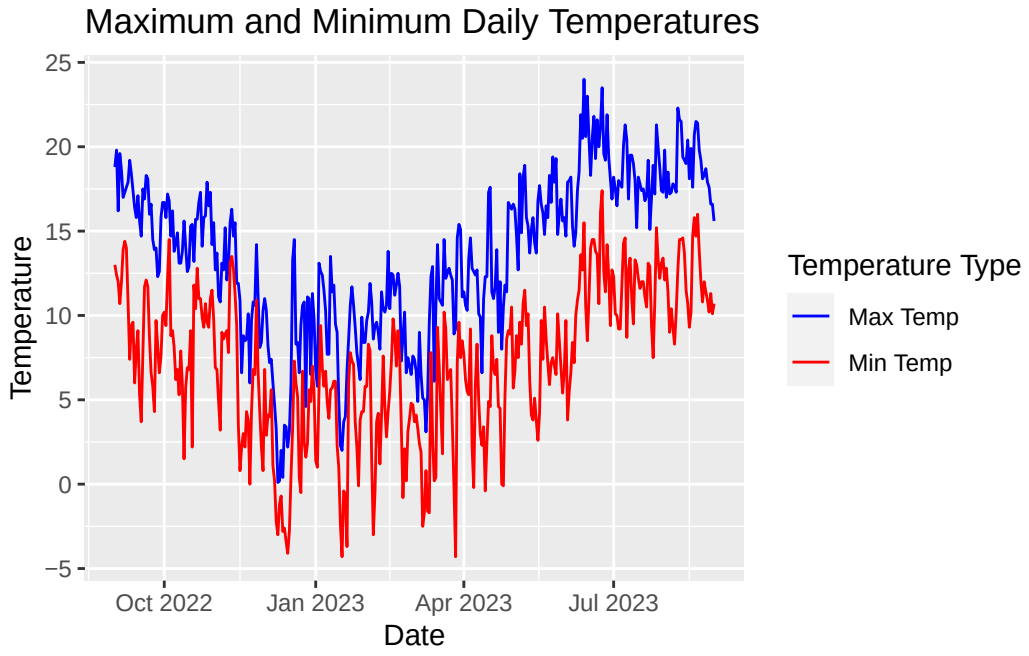
```
[1] Jul
12 Levels: Jan < Feb < Mar < Apr < May < Jun < Jul < Aug < Sep < ... < Dec
```

```r
print(lowest)
```

```
[1] Feb
12 Levels: Jan < Feb < Mar < Apr < May < Jun < Jul < Aug < Sep < ... < Dec
```

Q2: Use ggplot2 to create a time series plot of the maximum and minimum daily temperatures.
[The two time series must be on the same plot.]

```r
library(ggplot2)
library(dplyr)
library(tidyr)

max_daily_temp <- aggregate(dublin_bikes$air_temp, list(dublin_bikes$'date'), max) |> rena

min_daily_temp <- aggregate(dublin_bikes$air_temp, list(dublin_bikes$'date'), min) |> rena

daily_temp <- merge(max_daily_temp, min_daily_temp) # Merging

# Reshaping daily_temp to a long format using pivot_longer, so I can add legend
daily_temp_long <- daily_temp |>
  pivot_longer(cols = c(max_temp, min_temp), names_to = "Temperature_Type", values_to = "T

ggplot(daily_temp_long, aes(x = date, y = Temperature, color = Temperature_Type)) +
  geom_line() +
  labs(title = "Maximum and Minimum Daily Temperatures",
       x = "Date",
       y = "Temperature") +
  scale_color_manual(values = c("max_temp" = "blue", "min_temp" = "red"),
                     name = "Temperature Type",
                     labels = c("Max Temp", "Min Temp"))
```



Q3: Check if, according to this dataset, there has been on average more rain during the weekend (Sat-Sun) with respect to weekdays (Mon-Fri).

YES, there has been on average more rain during the weekend

```
daily_average <- dublin_bikes|>
  group_by(`day_of_week`) |>
  summarise(`average_rainfall` = mean(rainfall))

weekends_average <- daily_average |>
  filter(day_of_week %in% c("Sat", "Sun")) |>
  summarise(mean(average_rainfall))
weekends_average #0.14
```

```
# A tibble: 1 x 1
  `mean(average_rainfall)`
                     <dbl>
1                    0.141
```

```
weekdays_average <- daily_average |>
  filter(day_of_week %in% c("Mon", "Tues", "Wed", "Thu", "Fri")) |>
  summarise(mean(average_rainfall))
```

```
  weekdays_average #0.102
```

```
# A tibble: 1 x 1
  `mean(average_rainfall)`
                     <dbl>
1                    0.102
```

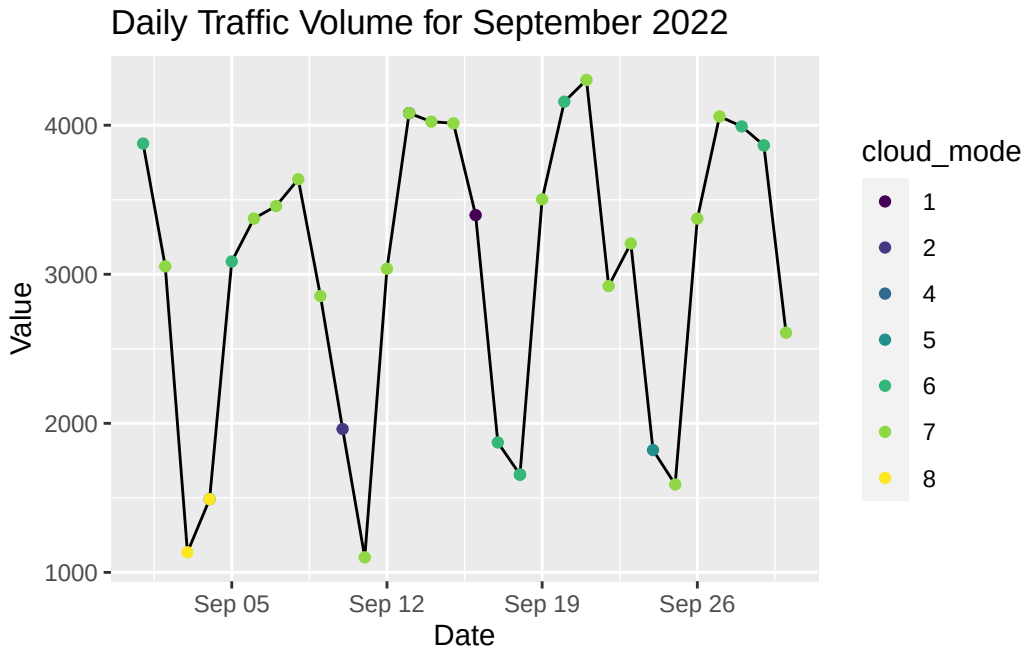
```
  names(dublin_bikes)
```

```
 [1] "date"                                "hour"
 [3] "day_of_week"                         "month"
 [5] "rainfall"                            "air_temp"
 [7] "wind_speed"                          "cloud_amount"
 [9] "Clontarf - James Larkin Rd"          "Clontarf - Pebble Beach Carpark"
[11] "Griffith Avenue (Clare Rd Side)"     "Griffith Avenue (Lane Side)"
[13] "Grove Road Totem"                    "Richmond Street Cyclists 1"
[15] "Richmond Street Cyclists 2"
```

Q4. Focus on the data for one month of the year of your choice, create a plot of the daily traffic volume in a locations of your choice, and the mode of the Cloud amount each day. Comment on your findings. [Notice that there isn't a built-in function to calculate the mode in R. The mode is defined as the most frequently occurring value in the set of observations.]

```
  library(dplyr)
  library(ggplot2)
  daily_traffic <- dublin_bikes |>
    filter(month == 'Sep') |>
    select(date, `Grove Road Totem`) |>
    group_by(date) |>
    summarise(daily_total = sum(`Grove Road Totem`))

  new_df <- dublin_bikes |>
    filter(month == 'Sep') |>
    select(`Grove Road Totem`, cloud_amount, date)


  cloud <- count(new_df, cloud_amount, date) |>
    group_by(date) |>
    filter(n == max(n)) |> #Filter to keep only rows where the count 'n' is equal to the max
    ungroup() |> #remove grouping
```

```
  select(cloud_mode = cloud_amount, date, n)

com_data <- merge(daily_traffic, cloud)

ggplot(com_data, aes(x=date, y=)) +
  geom_line(aes(y=daily_total)) +
  geom_point(aes(y=daily_total, color =cloud_mode)) +
  labs(title = "Daily Traffic Volume for September 2022", x = "Date", y = "Value")
```



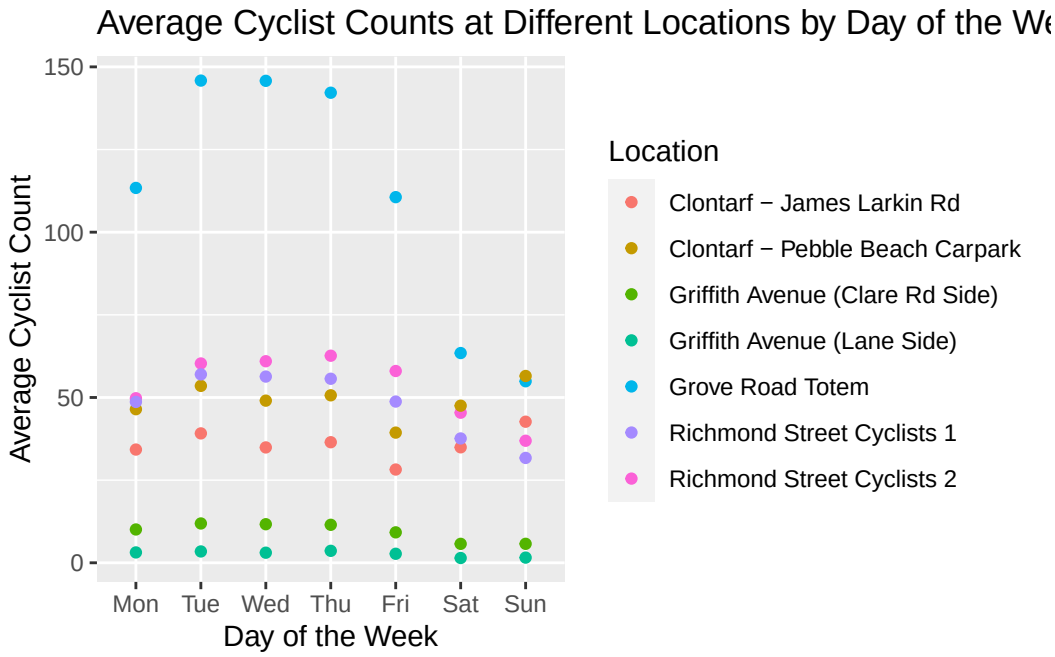
Task 3: Creativity

Do something interesting with these data! Create two plots or two tables or one plot and one table showing something we have not discovered above already and outline your findings.

From my plot, it appears that among the recorded areas, 'Grove Road Totem' generally experiences the heaviest traffic throughout the week. In contrast, 'Griffith Avenue (Lane Side)' and 'Griffith Avenue (Clare Rd Side)' are not as busy as the other locations recorded.

```
library(ggplot2)
library(dplyr)
library(tidyr)
```

```r
avg_counts <- aggregate(
    . ~ day_of_week,
    data = dublin_bikes[, c("day_of_week", "Clontarf - James Larkin Rd", "Griffith Avenue

# Reshape the data
avg_counts_long <- avg_counts |>
    pivot_longer(cols = -day_of_week, names_to = "Location", values_to = "Average Count")

ggplot(avg_counts_long, aes(x = day_of_week, y = `Average Count`, color = Location)) +
    geom_point() +
    labs(title = "Average Cyclist Counts at Different Locations by Day of the Week",
         x = "Day of the Week",
         y = "Average Cyclist Count")
```



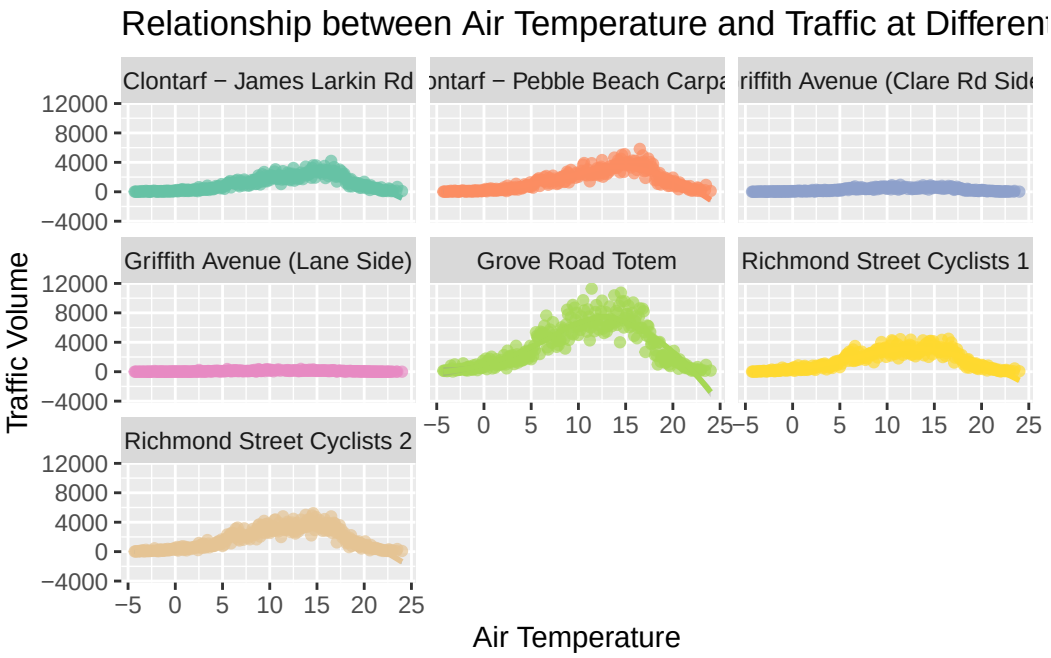
From the following plot, it appears that for both "Griffith Avenue (Lane Side)" and "Griffith Avenue (Clare Rd Side)," there is no discernible relationship between air temperature and bicycle traffic volume. This observation aligns with the previous graph, which indicated that these two locations experience the least amount of bicycle traffic compared to others. Consequently, temperature fluctuations do not significantly influence cycling activity in these areas. For the remaining locations, there seems to be a consistent peak in bicycle traffic when the temperature is between 10-15 degrees Celsius.

```r
temp_traffic <- dublin_bikes |>
  select("air_temp", "Clontarf - James Larkin Rd", "Griffith Avenue (Clare Rd Side)", "Gro
  group_by(air_temp) |>
  summarise(across(everything(), sum))

temp_traffic_long <- pivot_longer(temp_traffic,
                          cols = -air_temp,
                          names_to = "Location",
                          values_to = "Traffic")


ggplot(temp_traffic_long, aes(x = air_temp, y = Traffic, color = Location)) +
  geom_point(alpha = 0.7) +
  geom_smooth() +
  scale_color_brewer(palette = "Set2") +
  facet_wrap(~Location) +# Create a separate plot for each location
  guides(color = 'none') +
    labs(title = "Relationship between Air Temperature and Traffic at Different Locations"
       x = "Air Temperature",
       y = "Traffic Volume")
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
Warning: Removed 32 rows containing non-finite values (`stat_smooth()`).
```

```
Warning: Removed 32 rows containing missing values (`geom_point()`).
```


Relationship between Air Temperature and Traffic at Differen
Clontarf − James Larkin Rd
ontarf − Pebble Beach Carpa
riffith Avenue (Clare Rd Side
Griffith Avenue (Lane Side)
Grove Road Totem
Richmond Street Cyclists 1
Richmond Street Cyclists 2
Traffic Volume
12000
8000
4000
0
−4000
−5
0
5
10
15
20
25
Air Temperature