Uber Data Analysis Project - Assignment 4 ANA-515

Rob Brown

5/9/2021

**Executive Summary:**

This data analytics project practices code and concepts found in an Uber Data Analysis project on the Data Flair website. Practice of the R code can be found in an R script file on my GitHub site along with this R Markdown document and the datasets. This is done in fulfillment of Assignment #4 of McDaniel College’s ANA-515 course.

**Business Goal of the Project:**

This project performs some basic analysis and data visualizations of Uber data from April 2014 to September 2014 in New York City. Through this analysis, the business goal is to gain insights into the travel habits of these New York City Uber riders. Uber data sets are stored in a series of zipped csv files on a Google page, linked off of the Data Flair website.

As a part of the project, the following packages must be installed in R Studio: ggplot2, ggthemes, lubridate, dplyer, tidyr, DT, scales, and readr.

library(ggplot2)

Warning: package 'ggplot2' was built under R version 4.0.5

library(ggthemes)

Warning: package 'ggthemes' was built under R version 4.0.5

library(lubridate)

Warning: package 'lubridate' was built under R version 4.0.5

Attaching package: 'lubridate'

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

library(dplyr)

Warning: package 'dplyr' was built under R version 4.0.5

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(tidyr)

Warning: package 'tidyr' was built under R version 4.0.4

library(DT)

Warning: package 'DT' was built under R version 4.0.5

library(scales)

Warning: package 'scales' was built under R version 4.0.5

library(readr)

Warning: package 'readr' was built under R version 4.0.4

Attaching package: 'readr'

The following object is masked from 'package:scales':  
  
 col\_factor

**Set-up of Data Environment**

Set Working Directory:

The code below sets the working directory and imports the data. This code gets and establishes the working drive to be a specific folder on my personal computer.

getwd()

[1] "C:/Users/coach/OneDrive/Documents/Data Analytics/ANA-515 Data Storage, Retrieval and Preparation/Module 4"

setwd("C:\\Users\\coach\\OneDrive\\Documents\\Data Analytics\\ANA-515 Data Storage, Retrieval and Preparation\\Module 4")  
getwd()

[1] "C:/Users/coach/OneDrive/Documents/Data Analytics/ANA-515 Data Storage, Retrieval and Preparation/Module 4"

Color Palette:

This creates a vector of colors for later use in plotting.

colors=c("#CC1011","#665555","#05a399","#cfcaca","#f5e840","#0683c9","#e075b0")

Importing Data:

R Code below uses the read\_csv() command to import the data from each month’s csv file into an appropriately named variable. Afterwards, an rbind() command was used to create a single data\_2014 dataframe by combining together all of the monthly dataframes.

apr\_data <- read\_csv("uber\_raw\_data\_apr14.csv/uber-raw-data-apr14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

may\_data <- read\_csv("uber-raw-data-may14.csv/uber-raw-data-may14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

jun\_data <- read\_csv("uber-raw-data-jun14.csv/uber-raw-data-jun14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

jul\_data <- read\_csv("uber-raw-data-jul14.csv/uber-raw-data-jul14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

aug\_data <- read\_csv("uber-raw-data-aug14.csv/uber-raw-data-aug14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

sep\_data <- read\_csv("uber-raw-data-sep14.csv/uber-raw-data-sep14.csv")

-- Column specification --------------------------------------------------------  
cols(  
 `Date/Time` = col\_character(),  
 Lat = col\_double(),  
 Lon = col\_double(),  
 Base = col\_character()  
)

data\_2014 <- rbind(apr\_data,may\_data,jun\_data,jul\_data,aug\_data,sep\_data)

Initial Scan of Dataframe and Column Names:

View(data\_2014)  
colnames(data\_2014)

[1] "Date/Time" "Lat" "Lon" "Base"

**Data Preparation & Description**

The data\_2014 dataframe has 4534327 rows and 4 columns. Initially, the variables are Date/Time, Latitude, Longitude, and Base. The Date/Time variable contains both pieces of information combined into one variable; more details given below on formatting this column into additional time factors. The Latitude and Longitude give spatial coordinates on the Earth of these Uber trips. Bases gives a label of one of five base locations for these rides. Given the nature of all four variables, they do not lend themselves to summary statistics as none are quantitative variables. As such, primary analysis will occur from the data visualizations given below using a series of commands from ggplot2.

This block of code formats the Date/Time column into Month/Day/Year Hour/Minute/Second and then creates objects for day, month, year, day of week, hour, minute, and second. I’m noting here that this code differs from the Data Flair website slightly as they listed initial column as Date.Time when it appeared in the dataframe as Date/Time. When I typed in data\_2014 with the $ sign to get a specific column, I selected the Data/Time option and it appeared with the `` marks. With that said, it took a little while to figure that out after several errors in running the R code prior to that discovery. Code below uses the POSIXct() command. I used the colnames() command a few times to check my work along the way.

data\_2014$`Date/Time` <- as.POSIXct(data\_2014$`Date/Time`, format = "%m/%d/%Y %H:%M:%S")  
colnames(data\_2014)

[1] "Date/Time" "Lat" "Lon" "Base"

data\_2014$Time <- format(as.POSIXct(data\_2014$`Date/Time`, format = "%m/%d/%Y %H:%M:%S"), format="%H:%M:%S")  
colnames(data\_2014)

[1] "Date/Time" "Lat" "Lon" "Base" "Time"

data\_2014$`Date/Time` <- ymd\_hms(data\_2014$`Date/Time`)  
data\_2014$day <- factor(day(data\_2014$`Date/Time`))  
data\_2014$month <- factor(month(data\_2014$`Date/Time`, label = TRUE))  
data\_2014$year <- factor(year(data\_2014$`Date/Time`))  
data\_2014$dayofweek <- factor(wday(data\_2014$`Date/Time`, label = TRUE))  
data\_2014$hour <- factor(hour(hms(data\_2014$Time)))  
data\_2014$minute <- factor(minute(hms(data\_2014$Time)))  
data\_2014$second <- factor(second(hms(data\_2014$Time)))  
colnames(data\_2014)

[1] "Date/Time" "Lat" "Lon" "Base" "Time" "day"   
 [7] "month" "year" "dayofweek" "hour" "minute" "second"

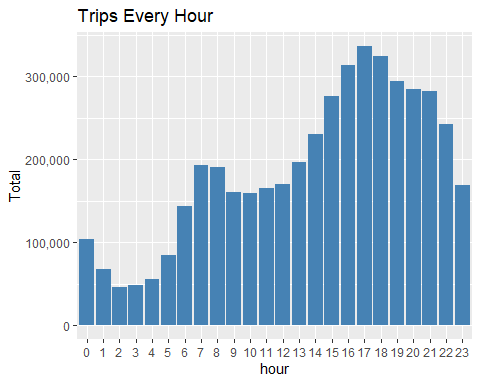
**Data Visualizations - Bar Charts**

**Trips Every Hour**

In this first bar chart, Uber trips are plotted by hours in a day (using a 24-hour clock) after creating an “hour\_data” object in R. The summarize function from the dplyr library is used to create a total. A data table is created and viewed from the “hour\_data” object.

hour\_data <- data\_2014 %>%  
 group\_by(hour) %>%  
 dplyr::summarize(Total = n())   
  
datatable(hour\_data)

View(hour\_data)  
  
ggplot(hour\_data, aes(hour, Total)) +   
 geom\_bar( stat = "identity", fill = "steelblue") +  
 ggtitle("Trips Every Hour") +  
 theme(legend.position = "none") +  
 scale\_y\_continuous(labels = comma)



The bar chart above has hours on the horizontal and totals on the vertical axes. The bars have a solid steelblue fill. Code from Data Flair also demonstrated a red outline to the bars as a feature of ggplot; I eliminated this option as the red outline contributes to visual clutter. In terms of analysis, the 5-6 pm hour has the most number of trips across all bases from April to September 2014. From a broader perspective, the number of Uber trips is at a minimum in the early morning hours from 2-4 am. This begins to increase from 4 am to about 8 am in the morning, drops off slightly at 9 am and holds constant to about 12 noon. Afterwards, the number of trips builds from 12 noon to the maximum at 5 pm. There is a slight decrease from 6 pm to 7 pm, holds constant from 7-9 pm, and then drops throughout the rest of the night into the early morning hours of 2-4 am. A reasonable assumption would that the two local maximums occur roughly during the morning and evening commuting times with work and school times.

**Trips by Hour and Month**

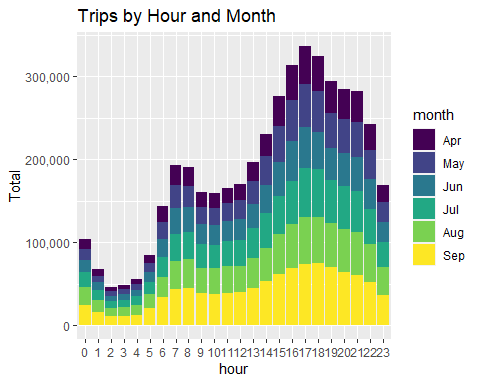
The code below creates a “month\_hour” object from data\_2014. Data is organized first by month, and then by the hour of the day for each month.

month\_hour <- data\_2014 %>%  
 group\_by(month, hour) %>%  
 dplyr::summarize(Total = n())

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

datatable(month\_hour)

View(month\_hour)  
  
ggplot(month\_hour, aes(hour, Total, fill = month)) +   
 geom\_bar( stat = "identity") +  
 ggtitle("Trips by Hour and Month") +  
 scale\_y\_continuous(labels = comma)



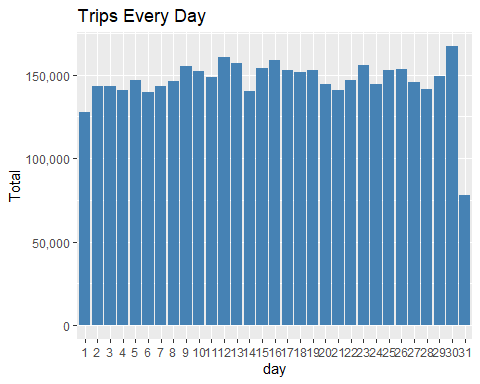
This creates a bar chart like before; however, the bars have monthly data superimposed onto them. While colorful, the bar chart is a little more difficult to analyze. What we can discern is that the trip behavior of New Yorkers is roughly the same over these six months across the 24-hour time frame. All months have similar local minimums and maximums in trip behavior, and all have similiar increases and decreases in number of trips at roughly the same times of the day.

**Trips Every Day**

This creates a “day\_group object” from data\_2014. This bar chart catalogs every day of the month on the horizontal axis with total number of trips on the vertical axis.

day\_group <- data\_2014 %>%  
 group\_by(day) %>%  
 dplyr::summarize(Total = n())   
datatable(day\_group)

View(day\_group)  
  
ggplot(day\_group, aes(day, Total)) +   
 geom\_bar( stat = "identity", fill = "steelblue") +  
 ggtitle("Trips Every Day") +  
 theme(legend.position = "none") +  
 scale\_y\_continuous(labels = comma)



Overall, this bar chart does not provide a lot of new information. The number of trips per day is roughly the same for an entire month. The minimum bar occurs on day 31, which is not surprising as only three of the six surveyed months has 31 days. Disregarding that bar, least number of Uber trips is on the first day of the month with the most number on day 30.

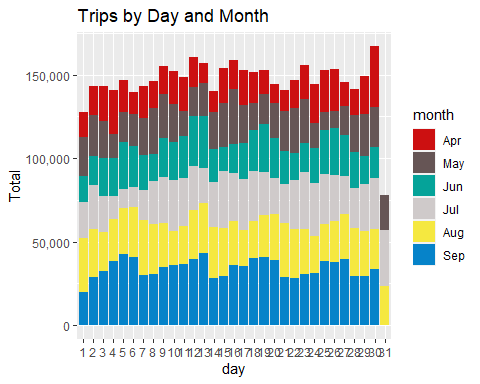
**Trips by Day and Month**

This creates a “day\_month\_group” object that further stratifies the prior plot. In this plot, the trips per days of the month are subdivided into the months.

day\_month\_group <- data\_2014 %>%  
 group\_by(month, day) %>%  
 dplyr::summarize(Total = n())

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

ggplot(day\_month\_group, aes(day, Total, fill = month)) +   
 geom\_bar( stat = "identity") +  
 ggtitle("Trips by Day and Month") +  
 scale\_y\_continuous(labels = comma) +  
 scale\_fill\_manual(values = colors)



While difficult to compare trips by day of the month across any of the months, what can be seen in the month of September is a repeating pattern where the number of Uber trips builds to a local maximum and then drops off immediately to a local minimum the next day. With additional time and data from 2014, additional analysis would be helpful to see if the local maximums correspond to a particular day of the week (Thursday or Friday to relax with friends) or if there is any correlation between the number of trips and holidays of this time period (Easter, Memorial Day, Fourth of July, Labor Day).

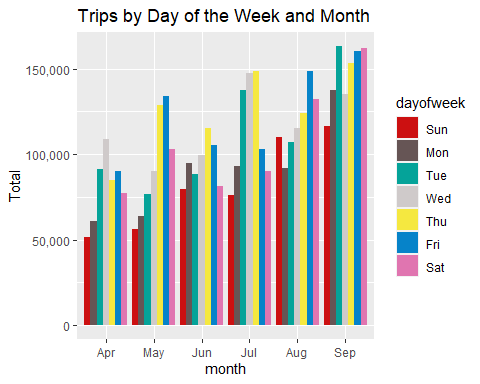
**Trips by Day of the Week and Month**

This creates a “month\_weekday object” from data\_2014. The bar graph puts a day of the week (by month) on the horizontal and totals on the vertical.

month\_weekday <- data\_2014 %>%  
 group\_by(month, dayofweek) %>%  
 dplyr::summarize(Total = n())

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

ggplot(month\_weekday, aes(month, Total, fill = dayofweek)) +   
 geom\_bar( stat = "identity", position = "dodge") +  
 ggtitle("Trips by Day of the Week and Month") +  
 scale\_y\_continuous(labels = comma) +  
 scale\_fill\_manual(values = colors)



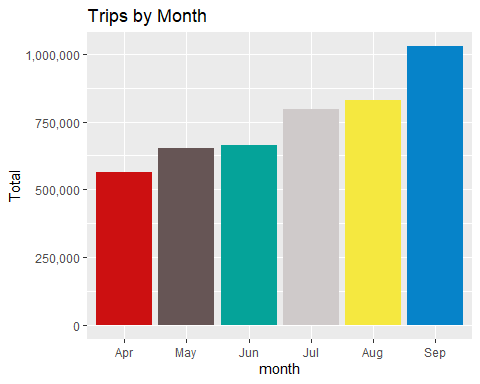
I left this graph in as an example of visual clutter. It is extremely difficult to pick out a day of the week and follow it across all of the months, given 7 colors. Overall, there is a trend of more Uber trips as the months go along. In addition, this graph does provide a little more context to the prior graph as we identified day of the week as an item of additional study. What is curious to me is that the maximum number of Uber trips by day of the week is different from month to month. For example, Wednesday has the most number of trips in April. In May, more trips were on Friday and closely followed up by Thursday. Thursday is the most popular day of the week for June and July, with Friday overtaking it for August. September has Tuesday, Thursday, Friday, and Saturday all equally popular. Additional analysis would be needed to parse out these reasons, such as Uber trips related to work obligations or vacationing.

**Trips by Month**

This creates a “month\_group object” from data\_2014. The bar chart has each of the months on the horizontal axis with trip totals on the vertical axis.

month\_group <- data\_2014 %>%  
 group\_by(month) %>%  
 dplyr::summarize(Total = n())   
datatable(month\_group)

ggplot(month\_group, aes(month, Total, fill = month)) +   
 geom\_bar( stat = "identity") +  
 ggtitle("Trips by Month") +  
 theme(legend.position = "none") +  
 scale\_y\_continuous(labels = comma) +  
 scale\_fill\_manual(values = colors)

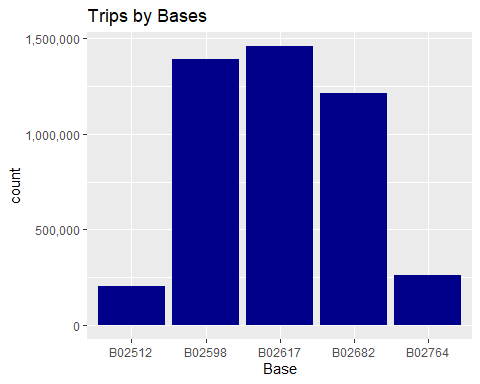


From this bar chart, we can see that the total number of Uber trips in New York City has increased from April to September. By heights of the bars, the biggest increase in Uber rides occurred in the month of September.

**Trips by Bases**

This section creates a visualization using the base label on the horizontal axis with trip totals on the vertical axis.

ggplot(data\_2014, aes(Base)) +   
 geom\_bar(fill = "darkblue") +  
 scale\_y\_continuous(labels = comma) +  
 ggtitle("Trips by Bases")

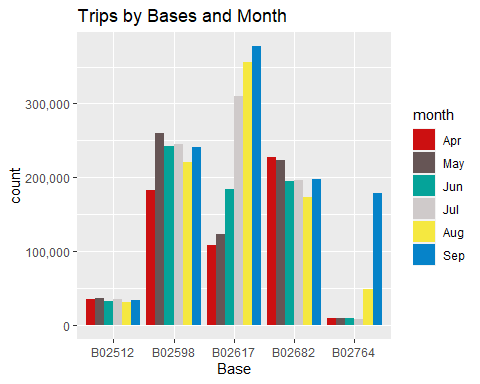


Overall, there are 5 bases. Base B02617 had the most number of Uber trips followed by B02598. Base B02512 had the least number of trips.

**Trips be Bases and Month**

This plot has bases on the horizontal, separated by month.

ggplot(data\_2014, aes(Base, fill = month)) +   
 geom\_bar(position = "dodge") +  
 scale\_y\_continuous(labels = comma) +  
 ggtitle("Trips by Bases and Month") +  
 scale\_fill\_manual(values = colors)

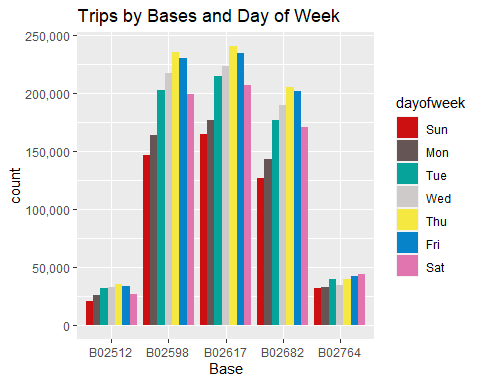


Base B02617 had the most number of Uber trips in the months of July, August, and September as compared to all of the other months of all the other bases. Base B02598 (identified in the last bar chart with the second most trips) had its most number of trips in the month of May. Base B02512 had pretty consistent number of Uber trips across all six months. Base B02764 had very few trips from April to July, a slight jump in August, and then a very dramatic increase in September. If more data was supplied, it would be interesting to see how the number of trips for this base in successive months compared to these six months.

**Trips by Bases and Day of the Week**

This plot has the trips per base separated by day of the week.

ggplot(data\_2014, aes(Base, fill = dayofweek)) +   
 geom\_bar(position = "dodge") +  
 scale\_y\_continuous(labels = comma) +  
 ggtitle("Trips by Bases and Day of Week") +  
 scale\_fill\_manual(values = colors)



Again, this type of chart is attempting to provide more in-depth analysis; however, the number of colors is challenging for the reader. For four out of the five bases, Thursday is the most popular day of the week for an Uber, followed by Friday.

**Heatmap by Hour and Day**

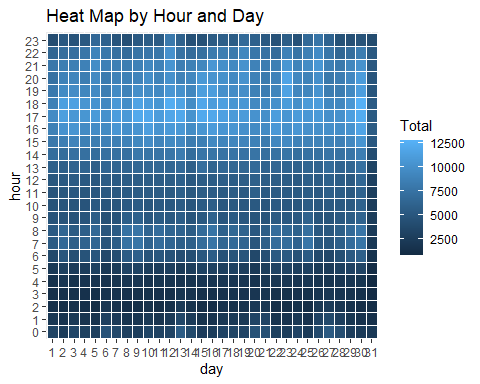
This section creates a heatmap using ggplot. On this particular heatmap, day of the month is on the horizontal axis with hour of the day on the vertical axis. While I will create them below and they are colorful, I find them a little difficult to read. Using this legend, the lighter the color, the more Uber trips.

day\_and\_hour <- data\_2014 %>%  
 group\_by(day, hour) %>%  
 dplyr::summarize(Total = n())

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

datatable(day\_and\_hour)

ggplot(day\_and\_hour, aes(day, hour, fill = Total)) +  
 geom\_tile(color = "white") +  
 ggtitle("Heat Map by Hour and Day")

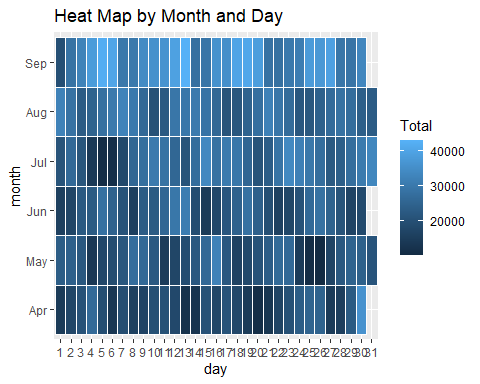


Overall, it appears that the lightest colors are around hours 16-18, or 4-6 pm. This probably corresponds to many folks leaving work at that time.

**Heatmap by Month and Day**

This heatmap continues with day of the month on the horizontal but uses month of the year on the vertical axis.

ggplot(day\_month\_group, aes(day, month, fill = Total)) +  
 geom\_tile(color = "white") +  
 ggtitle("Heat Map by Month and Day")

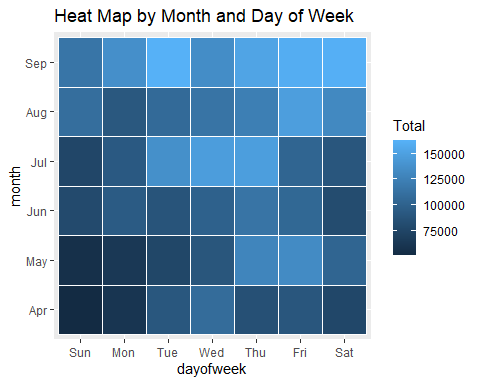


This visualization shows lighter colors in the month of September, meaning that there were more Uber trips in this month than in other months. Likewise, April and May has the darkest colors, corresponding to fewer trips.

**Heatmap by Month and Day of the Week**

This heatmap plots day of the week on the horizontal axis with months on the vertical axis. This has the same type of shading; lighter blues indicate more Uber trips than darker blues.

ggplot(month\_weekday, aes(dayofweek, month, fill = Total)) +  
 geom\_tile(color = "white") +  
 ggtitle("Heat Map by Month and Day of Week")



The least number of trips was on Sundays and Mondays in the months of April and May. The most number of trips was taken in the month of September. In the month of July, the most popular days were Tuesday through Thursday. Friday was the most popular day for August.

**Heatmap by Month and Bases**

This heatmap plotted bases on the horizontal axis with months on the vertical axis.

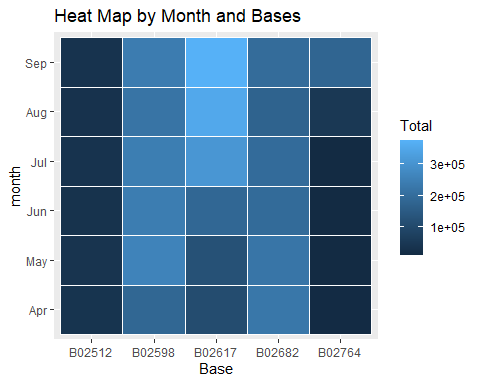
month\_base <- data\_2014 %>%  
 group\_by(Base, month) %>%  
 dplyr::summarize(Total = n())

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

day0fweek\_bases <- data\_2014 %>%  
 group\_by(Base, dayofweek) %>%  
 dplyr::summarize(Total = n())

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

ggplot(month\_base, aes(Base, month, fill = Total)) +  
 geom\_tile(color = "white") +  
 ggtitle("Heat Map by Month and Bases")

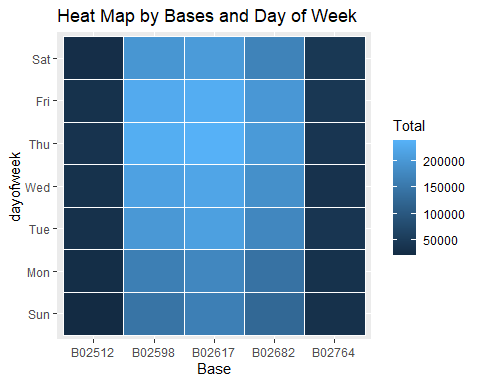


As noted in the other visualizations, base B02617 has the most number of trips as compared to the other bases. This particular base had strong ridership for the months July through September. Base B02512 had the least number of trips (darkest blues) followed by base B02764.

**Heatmap by Bases and Day of the Week.**

This final heatmap plotted bases by days of the week.

ggplot(day0fweek\_bases, aes(Base, dayofweek, fill = Total)) +  
 geom\_tile(color = "white") +  
 ggtitle("Heat Map by Bases and Day of Week")



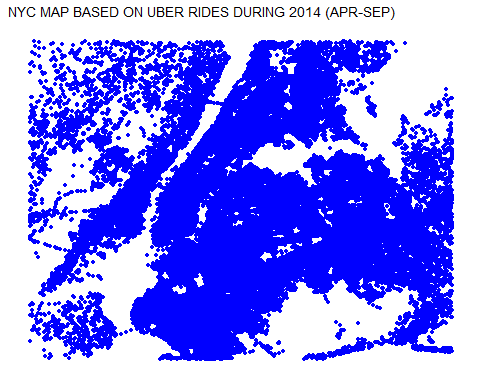
This heatmap shows a progression to lighter blues during the week for bases B02598, B02617, and B02682. There is little differentiation for bases B02512 and B02764 with respect to days of the week.

**Map Visualization**

This section created a map visualization of the rides in New York with a geo-plot. I chose to only plot one of these as it took a very long time to generate.

min\_lat <- 40.5774  
max\_lat <- 40.9176  
min\_long <- -74.15  
max\_long <- -73.7004  
  
ggplot(data\_2014, aes(x=Lon, y=Lat)) +  
 geom\_point(size=1, color = "blue") +  
 scale\_x\_continuous(limits=c(min\_long, max\_long)) +  
 scale\_y\_continuous(limits=c(min\_lat, max\_lat)) +  
 theme\_map() +  
 ggtitle("NYC MAP BASED ON UBER RIDES DURING 2014 (APR-SEP)")

Warning: Removed 71701 rows containing missing values (geom\_point).



**Conclusion** This data analytics project was meant to practice R code from the Data Flair website, especially with regard to data visualizations available to us from ggplot2. In this project, only four variables was given to us to use, but quite a few visualizations were given throughout the project. Practice was given in formatting and separating date and time before any visualizations were generated. Graphing was done with bar charts, heatmaps, and one map visualization with some basic analysis achieved by reading these graphs.