Uber Pickups in New York City

Neftali Lemus Martinez

2023-04-07

About the Data:

The original directory for this project is composed of four groups of files that cover service statistics of Uber and non-Uber trips between 2014 and 2015. Each dataset provides different transit aspects of the mentioned services.

Link: https://www.kaggle.com/datasets/fivethirtyeight/uber-pickups-in-new-york-city

The Data

- Uber trip data from 2014 (April - September), separated by month, with detailed location information.
- Uber trip data from 2015 (January - June), with less fine-grained location information.
- Non-Uber FHV (For-Hire Vehicle) trips. The trip information varies by company, but can include day of trip, time of trip, pickup location, driver's for-hire license number, and vehicle's for-hire license number.
- Aggregate ride and vehicle statistics for all FHV companies (and, occasionally, for taxi companies).

I will focus on the first group that contains detailed location information of Uber pickups in 2014.

Project Goal:

The following visualization project pertains over 4.5 million Uber pickups in New York City from April to September 2014. The goal is to dissect and understand user trends in one of the largest metropolitan cities in the world. The data insights can be further used to develop an in depth analysis of New York's transit system and the socioeconomic challenges with the rising Gig driven economy.

Part 1: Data Preparation

```
# Libraries
library(ggplot2)
library(tidyverse)
library(lemon)
library(DT)
library(scales)
library(mapview)

# Read in Data (April-Sep 2014)
apr <- read.csv("uber-raw-data-apr14.csv")
may <- read.csv("uber-raw-data-may14.csv")
jun <- read.csv("uber-raw-data-jun14.csv")
jul <- read.csv("uber-raw-data-jul14.csv")
aug <- read.csv("uber-raw-data-aug14.csv")
sep <- read.csv("uber-raw-data-sep14.csv")

# Raw Structure
str(apr)
```

```
## 'data.frame':    564516 obs. of  4 variables:
##  $ Date.Time: chr  "4/1/2014 0:11:00" "4/1/2014 0:17:00" "4/1/2014 0:21:00" "4/1/2014 0:28:00" ...
##  $ Lat      : num  40.8 40.7 40.7 40.8 40.8 ...
##  $ Lon      : num  -74 -74 -74 -74 -74 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

```
str(may)
```

```
## 'data.frame':    652435 obs. of  4 variables:
##  $ Date.Time: chr  "5/1/2014 0:02:00" "5/1/2014 0:06:00" "5/1/2014 0:15:00" "5/1/2014 0:17:00" ...
##  $ Lat      : num  40.8 40.7 40.7 40.7 40.8 ...
##  $ Lon      : num  -74 -74 -74 -74 -74 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

```
str(jun)
```

```
## 'data.frame':    663844 obs. of  4 variables:
##  $ Date.Time: chr  "6/1/2014 0:00:00" "6/1/2014 0:01:00" "6/1/2014 0:04:00" "6/1/2014 0:04:00" ...
##  $ Lat      : num  40.7 40.7 40.3 40.8 40.7 ...
##  $ Lon      : num  -74 -74 -74.7 -74 -74.2 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

```
str(jul)
```

```
## 'data.frame':    796121 obs. of  4 variables:
##  $ Date.Time: chr  "7/1/2014 0:03:00" "7/1/2014 0:05:00" "7/1/2014 0:06:00" "7/1/2014 0:09:00" ...
##  $ Lat      : num  40.8 40.8 40.7 40.8 40.7 ...
##  $ Lon      : num  -74 -74 -74 -74 -74 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

```
str(aug)
```

```
## 'data.frame':    829275 obs. of  4 variables:
##  $ Date.Time: chr  "8/1/2014 0:03:00" "8/1/2014 0:09:00" "8/1/2014 0:12:00" "8/1/2014 0:12:00" ...
##  $ Lat      : num  40.7 40.7 40.7 40.7 40.7 ...
##  $ Lon      : num  -74 -74 -74.1 -74 -74 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

```
str(sep)
```

```
## 'data.frame':    1028136 obs. of  4 variables:
##  $ Date.Time: chr  "9/1/2014 0:01:00" "9/1/2014 0:01:00" "9/1/2014 0:03:00" "9/1/2014 0:06:00" ...
##  $ Lat      : num  40.2 40.8 40.8 40.7 40.8 ...
##  $ Lon      : num  -74 -74 -74 -74 -73.9 ...
##  $ Base     : chr  "B02512" "B02512" "B02512" "B02512" ...
```

By looking at the initial structure we know the following:

- Each file is composed of 4 variables; Date/Time (chr), Latitude (num), Longitude(num), and Base (chr).
- There is an ordered increase in Uber trips by month and the number of observations is balanced.
- We can safely bind the information to have a complete analysis of the service months.

Part 2: Data Cleaning

```
# Combining Data
data_2014 <- rbind(apr, may, jun, jul, aug, sep)
# NA Check
sum(is.na(data_2014))
```

```
## [1] 0
```

```
# Simplifying time frames
data_2014$Date.Time <- as.POSIXct(data_2014$Date.Time, format = "%m/%d/%Y %H:%M:%S")
data_2014$Time <- format(as.POSIXct(data_2014$Date.Time, format = "%m/%d/%Y %H:%M:%S"), format="%H:%M:%
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)))
head(data_2014,3)
```

```
##             Date.Time     Lat      Lon   Base     Time day month year dayofweek
## 1 2014-04-01 00:11:00 40.7690 -73.9549 B02512 00:11:00   1   Apr 2014       Tue
## 2 2014-04-01 00:17:00 40.7267 -74.0345 B02512 00:17:00   1   Apr 2014       Tue
## 3 2014-04-01 00:21:00 40.7316 -73.9873 B02512 00:21:00   1   Apr 2014       Tue
##   hour minute second
## 1    0     11      0
## 2    0     17      0
## 3    0     21      0
```

After binding the data, we have no missing observations.

Note : This is often not the case with modern data collection as there is no time for companies to revise millions of incoming daily records.

I have opted to break-down the Date/Time variable into different time frames to understand user trends in detail. By creating these new variables there are many questions arise, for example:

1. What are the busiest service hours for Uber drivers?
2. Does the day of the week influence the amount of pick ups?
3. What are the New York City areas with the most pick up requests?
4. What are the main differences between the months with the highest and lowest pickup request?

There are also other questions/ideas that can could inspire other projects (Predictive Modeling, Recommendation systems, Classification, etc.)

1. What external factors could have an influence in the amount of pickup requests?
2. Is there a linear relationship between the socioeconomic status of a New York City area and the amount of Uber pickup requests?
3. Has the New York's transit system been affected with the rise of independent transportation services like Uber?
4. What are the main differences between taking a traditional cab compared to requesting an Uber ride?
5. How many times does the average New Yorker requests an Uber in a Year?

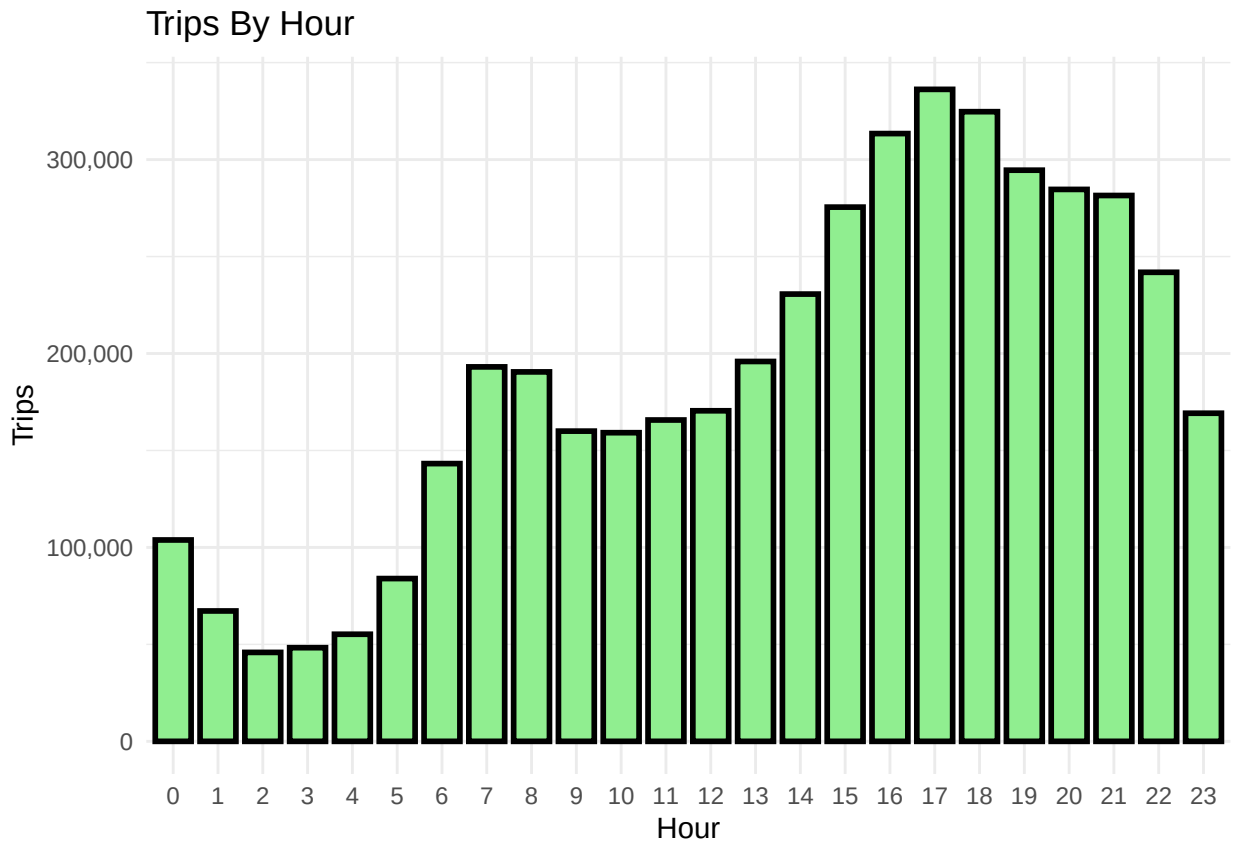
Part 3: Explanatory Data Analysis

Frequency of Pickups By Hour

```
options(scipen = 999)

trips_hour <- data_2014 %>%
  group_by(hour) %>%
  summarise(total = n())
```

```
ggplot(trips_hour, aes(x = hour, y = total)) +
  geom_bar(stat = "identity", fill = "lightgreen", color = "black", width = 0.8,cex = 1 ) +
  ggtitle("Trips By Hour") +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma)+
  xlab("Hour") +
  ylab("Trips")
```




```
# Table
head(trips_hour %>% arrange(desc(total)), 5)
```

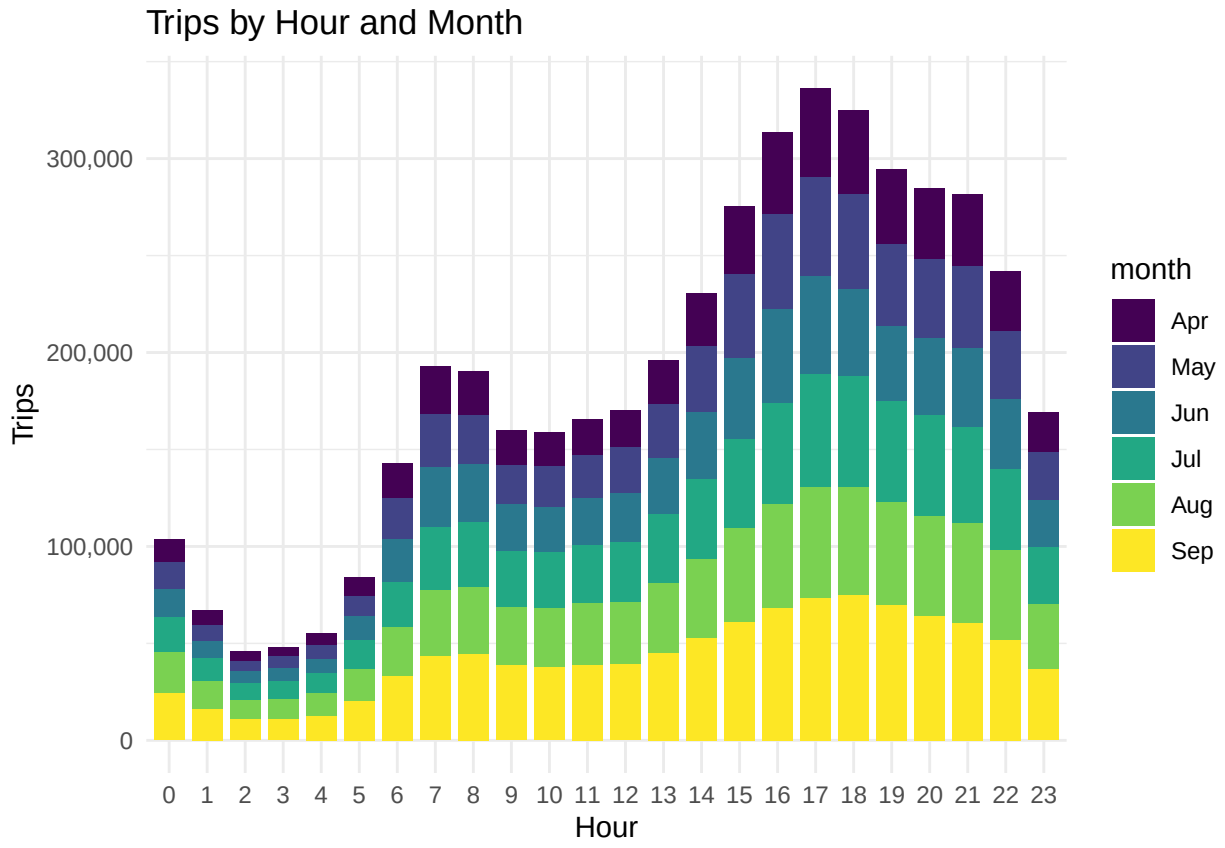
```
## # A tibble: 5 x 2
##   hour   total
##   <fct>  <int>
## 1 17    336190
## 2 18    324679
## 3 16    313400
## 4 19    294513
## 5 20    284604
```

The bar graph shows that the number of total number trips peaks between 4pm and 7pm, all over 290,000 requests per hour. On the other side, pickup requests were lowest between 1am-4am as they oscillate around 60,000 requests.

Frequency of Pickups by Month

```
month_hour <- data_2014 %>%
  group_by(month, hour) %>%
  summarise(Total = n(), .groups = "keep")

ggplot(month_hour, aes(hour, Total, fill = month)) +
  geom_bar( stat = "identity", width = 0.8) +
  theme_minimal() +
  ggtitle("Trips by Hour and Month") +
  scale_y_continuous(labels = comma) +
  xlab("Hour") +
  ylab("Trips")
```

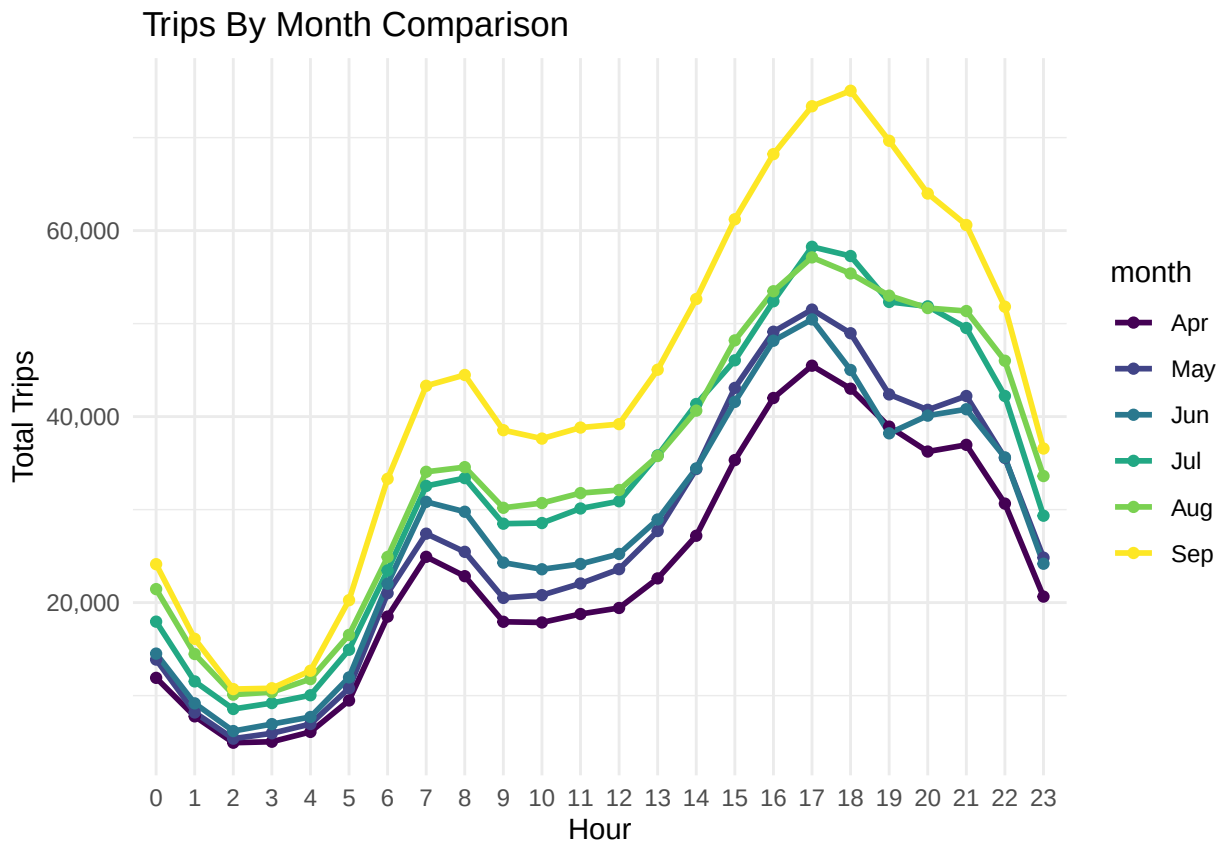


The monthly stacked Bar chart shows similar user pickup trends per month. Lets take a closer look at the numbers!

Montly Trip Comparison

```
ggplot(month_hour, aes(x=hour, y=Total, group = month, color = month)) +
  geom_line(linewidth = 1)+
  geom_point() +
  theme_minimal() +
```

```
  ggtitle(" Trips By Month Comparison") +
  scale_y_continuous(labels = comma) +
  xlab("Hour") +
  ylab("Total Trips")
```



Busiest vs Slowest Month

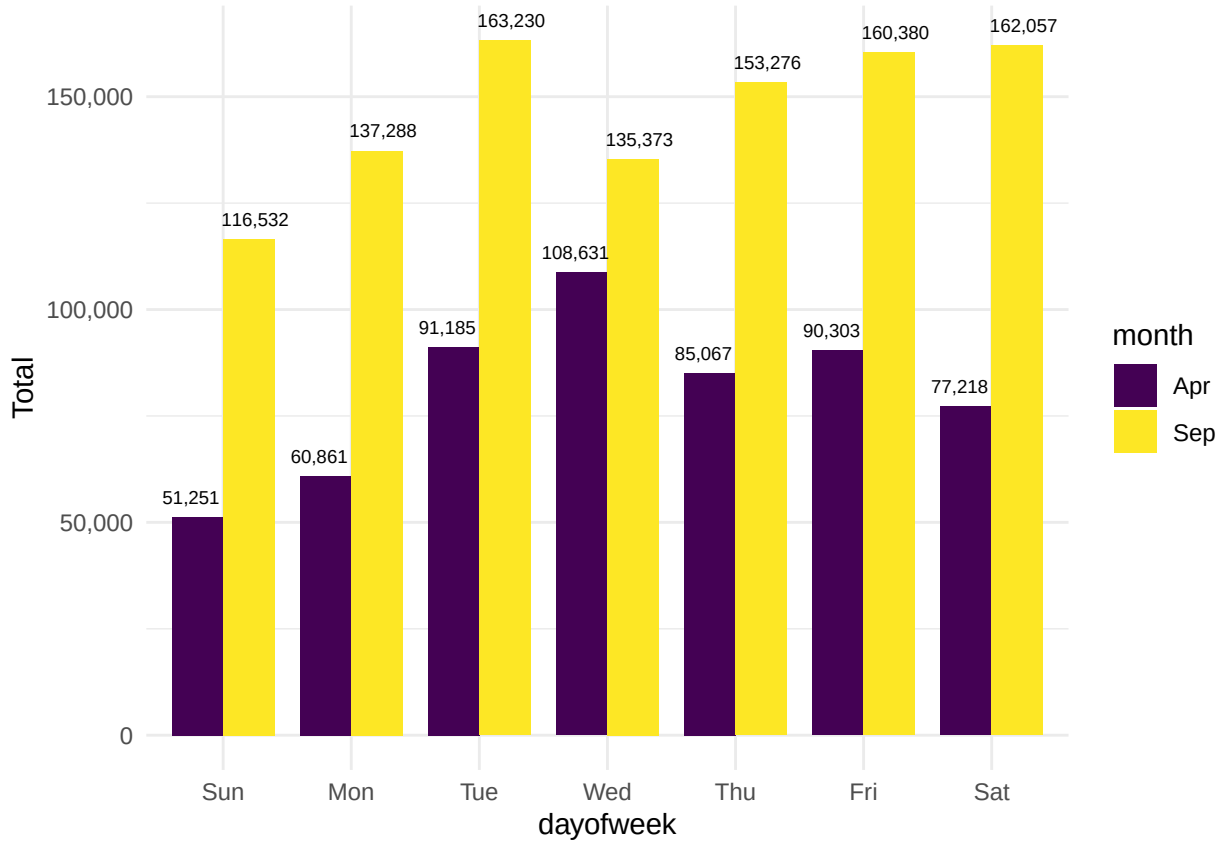
```
# Lets compare trip stats (min/max #trips)
month_weekday <- data_2014 %>%
   group_by(month, dayofweek) %>%
   dplyr::summarize(Total = n())
```

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

```
table(data_2014$month) # Max vs Min
```

```
##
##     Apr     May     Jun     Jul     Aug     Sep
##  564516  652435  663844  796121  829275 1028136
```

```
data2 <- month_weekday[c(1:7, 36:42),]
ggplot(data2, aes(dayofweek, y = Total, fill = month)) +
  geom_bar(stat ="identity",
           position = position_dodge(),
           width = 0.8) +
  geom_text(aes(label=comma(Total)),
            vjust=-1,
            color="black",
            cex=2.5,
            position = position_dodge(1), size=2.5) +
  theme_minimal() +
  scale_y_continuous(labels = comma)
```




```
comparison_num <- data2 %>% spread(key = month, value = Total) %>% mutate(Trip_Difference = Sep - Apr)
comparison_num
```

```
## # A tibble: 7 x 4
##   dayofweek    Apr    Sep Trip_Difference
##   <ord>      <int>  <int>           <int>
## 1 Sun        51251 116532           65281
## 2 Mon        60861 137288           76427
## 3 Tue        91185 163230           72045
## 4 Wed       108631 135373           26742
## 5 Thu        85067 153276           68209
## 6 Fri        90303 160380           70077
```

```
## 7 Sat        77218 162057           84839
```

```
summary(comparison_num)
```

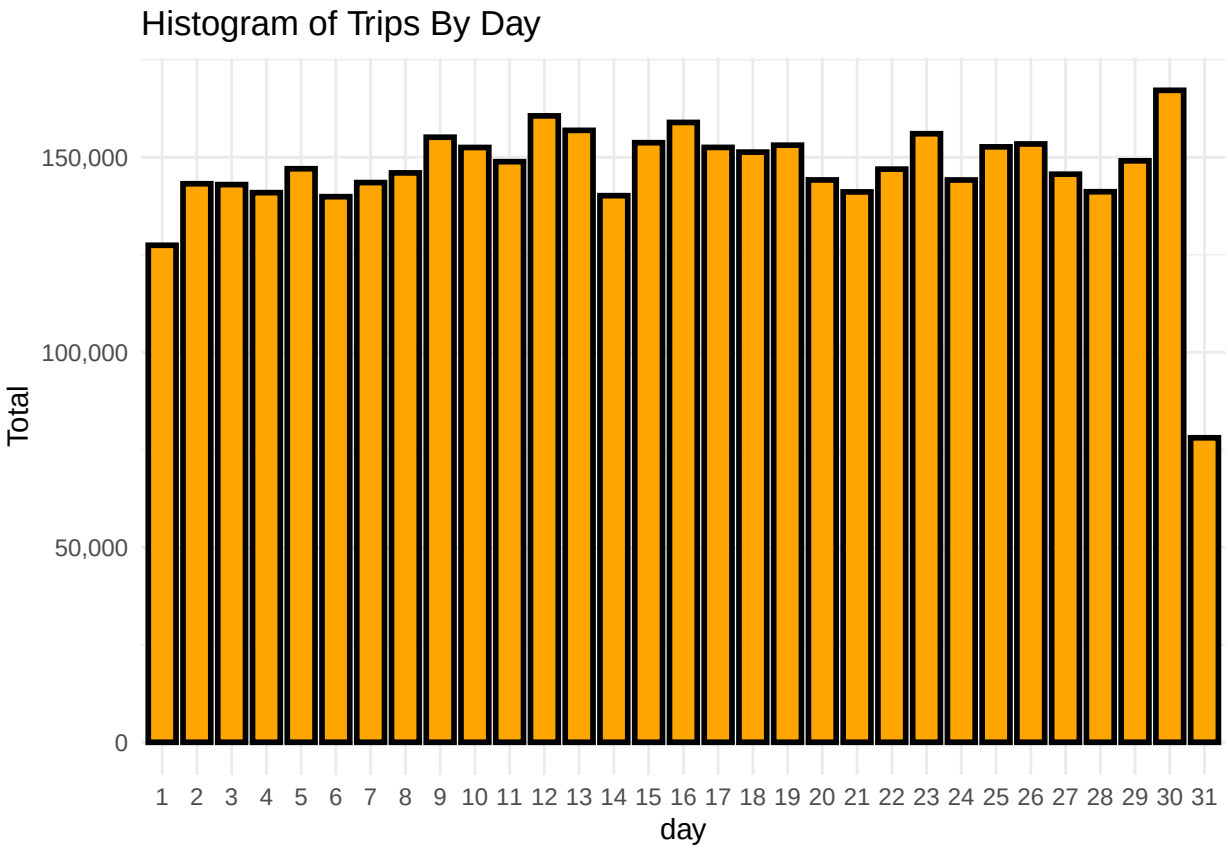
```
##  dayofweek      Apr              Sep         Trip_Difference
##  Sun:1     Min.   : 51251   Min.   :116532   Min.   :26742
##  Mon:1     1st Qu.: 69040   1st Qu.:136331   1st Qu.:66745
##  Tue:1     Median : 85067   Median :153276   Median :70077
##  Wed:1     Mean   : 80645   Mean   :146877   Mean   :66231
##  Thu:1     3rd Qu.: 90744   3rd Qu.:161219   3rd Qu.:74236
##  Fri:1     Max.   :108631   Max.   :163230   Max.   :84839
##  Sat:1
```

- The frequency table says shows April as the slowest month with 564,516 trips and September as the busiest with 1,028,136 trips.
- September has on average 66,231 trips more than April.

Histogram of Trips By Day of the Month

```
day_group <- data_2014 %>%
  group_by(day) %>%
  dplyr::summarize(Total = n())

# Histogram of Trips By day
ggplot(day_group, aes(day, Total)) +
  geom_histogram( stat = "identity", fill = "orange", color = "black", width = 0.8, cex =1) +
  ggtitle("Histogram of Trips By Day") +
  theme_minimal() +
  scale_y_continuous(labels = comma)
```

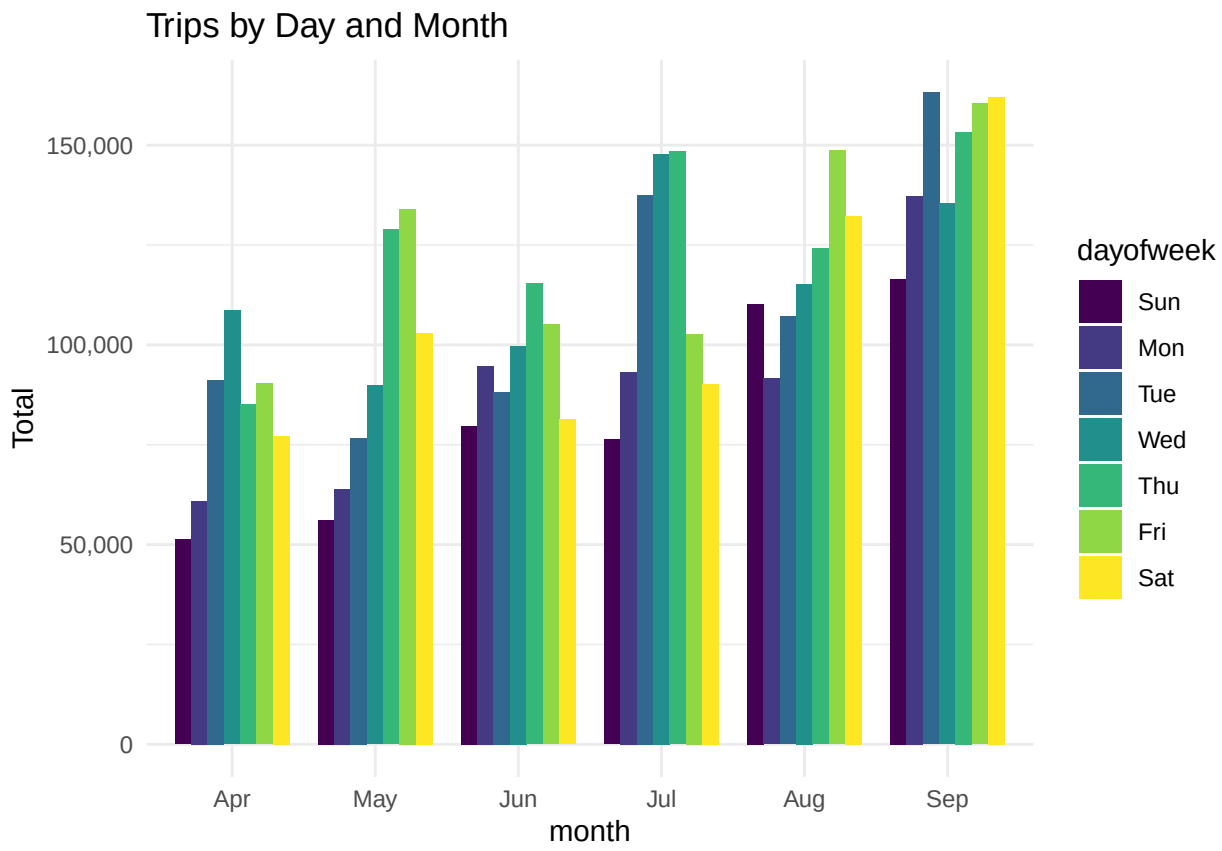


- The uniformity in the previous histogram shows that daily Uber pickups follow a normal distribution.

Trips By Day of the Week

```
month_weekday <- data_2014 %>%
  group_by(month, dayofweek) %>%
  dplyr::summarize(Total = n())

ggplot(month_weekday, aes(month, Total, fill = dayofweek)) +
  geom_bar( stat = "identity", position = "dodge", width = 0.8) +
  theme_minimal() +
  ggtitle("Trips by Day and Month") +
  scale_y_continuous(labels = comma)
```




```
data3 <- month_weekday %>% spread(key = month, value = Total) %>% mutate(Total_Trips = Apr + May + Jun
data3
```

```
## # A tibble: 7 x 8
##   dayofweek    Apr    May    Jun    Jul    Aug    Sep Total_Trips
##   <ord>      <int>  <int>  <int>  <int>  <int>  <int>       <int>
## 1 Sun        51251  56168  79656  76327 110246 116532      490180
## 2 Mon        60861  63846  94655  93189  91633 137288      541472
## 3 Tue        91185  76662  88134 137454 107124 163230      663789
## 4 Wed       108631  89857  99654 147717 115256 135373      696488
## 5 Thu        85067 128921 115325 148439 124117 153276      755145
## 6 Fri        90303 133991 105056 102735 148674 160380      741139
## 7 Sat        77218 102990  81364  90260 132225 162057      646114
```

```
# Summary of Total Trips
summary(data3$Total_Trips)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  490180  593793  663789  647761  718814  755145
```

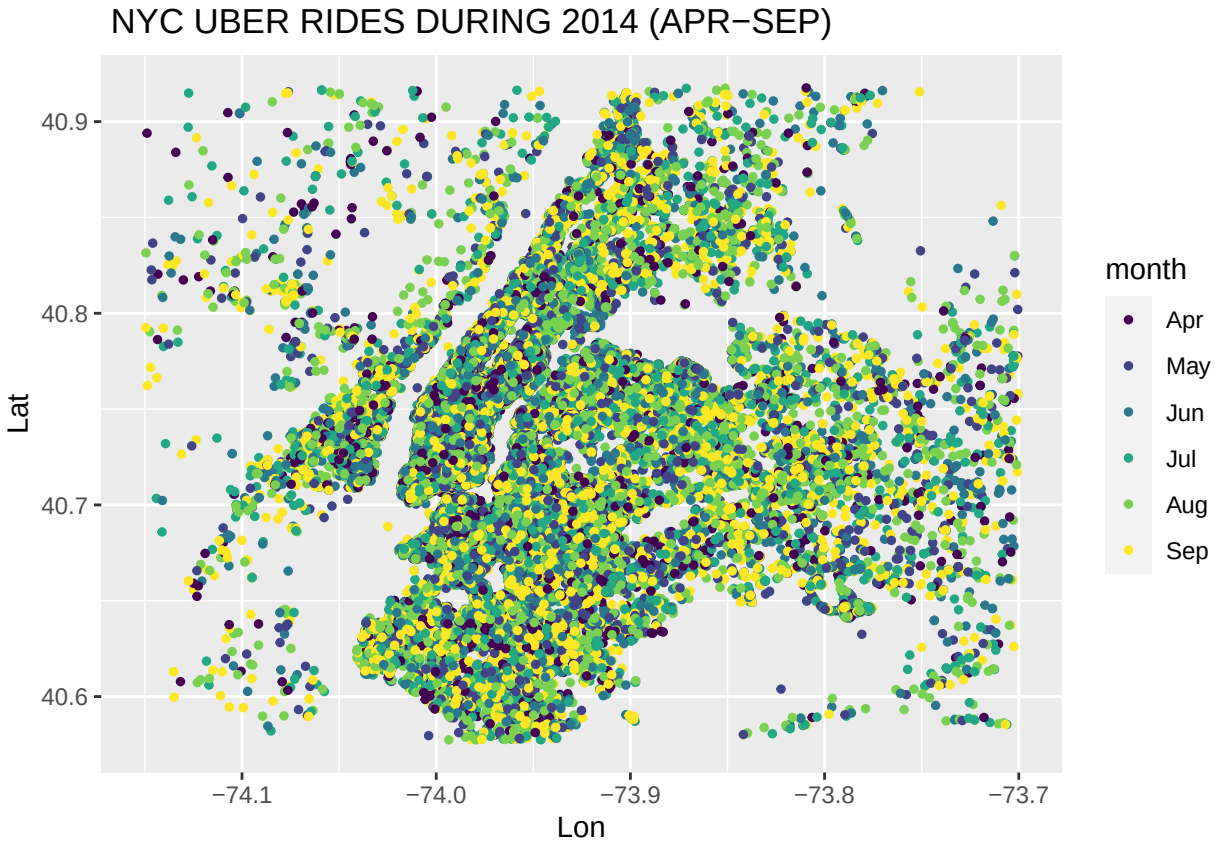
- Thursdays had the most trips while Sundays the least.
- There were on average 647,761 Uber pickups per month.

Part 4: Spatial Data

For the spatial visualization I opted to take a random sample of 400,000 out of the available 4.5 million, about 10% of the total information.

```
# Random sample (500,000 only)
rsamp <- sample.int(nrow(data_2014), size = 400000, replace = F)
map_data <- data_2014[rsamp,]
min_lat <- 40.5774
max_lat <- 40.9176
min_long <- -74.15
max_long <- -73.7004

ggplot(map_data, aes(x=Lon, y=Lat, color = month)) +
  geom_point(size=1) +
     scale_x_continuous(limits=c(min_long, max_long)) +
      scale_y_continuous(limits=c(min_lat, max_lat)) +
           ggtitle(" NYC UBER RIDES DURING 2014 (APR-SEP)")
```



Conclusion

This quick visualization project analyzed over 4.5 million Uber requests during a 6-month period in New York City. By grouping, transforming, and simplifying the respective records, we were able to uncover multiple user trends when it comes to ordering a ride. In short, we were able to know the busiest service hours, day's

with the most pickups, service requests by month and the areas with the most service requests. This type of information can serve as a starting point for many potential projects towards a better understanding of the socioeconomic effects brought the seasonal influx of people in the largest metropolitan city in the world.

Take it Further By

- Looking into the seasonal weather at NYC during this time period.
- Looking into the use of public transportation in New York City.
- Simplifying the data by focusing on specific areas of the city (e.g. Queens, Brooklyn).
- Comparing and contrasting Uber trends with Non-Uber ride services (group file 3).
- Looking into NYC airport data.
- Analyzing rice comparisons between services.
- Analyzing New York City's Traffic Trends.
- If possible, grouping requests by User ID.