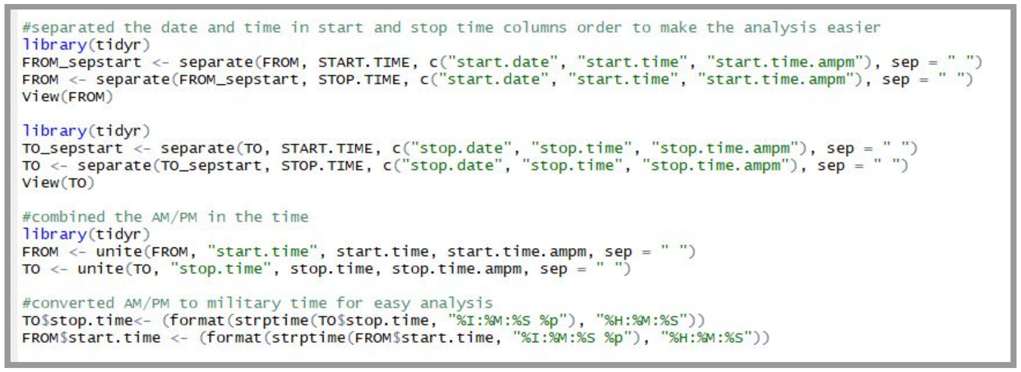
Divvy Capstone: Wrangling Milestone

Introduction

The purpose of this first milestone was to properly wrangle my dataset in order to get an improved overview on the allocation of observations by day. The first sections of this analysis involved formatting the data properly, concentrating on date and time. The second section covers double checking for any NA values and making sure the number of observations in my post-wrangling datasets matched the original raw data. Finally, I visualized the data by year and month and was able to find a pattern of seasonality.

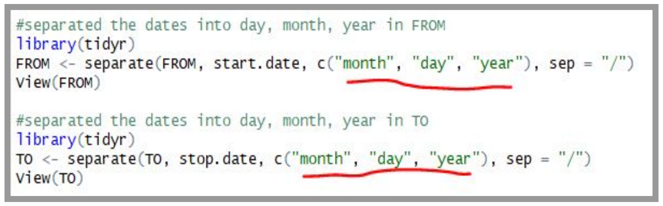
1. Tidy up name columns

- Date and time were originally combined, I separated the 2 and converted times t military time, waited to convert dates to Date in R so I would be able to group them



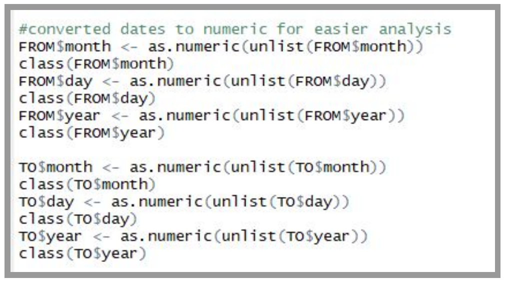
```r
#separated the date and time in start and stop time columns order to make the analysis easier
library(tidyr)
FROM_sepstart <- separate(FROM, START.TIME, c("start.date", "start.time", "start.time.ampm"), sep = " ")
FROM <- separate(FROM_sepstart, STOP.TIME, c("start.date", "start.time", "start.time.ampm"), sep = " ")
View(FROM)

library(tidyr)
TO_sepstart <- separate(TO, START.TIME, c("stop.date", "stop.time", "stop.time.ampm"), sep = " ")
TO <- separate(TO_sepstart, STOP.TIME, c("stop.date", "stop.time", "stop.time.ampm"), sep = " ")
View(TO)

#combined the AM/PM in the time
library(tidyr)
FROM <- unite(FROM, "start.time", start.time, start.time.ampm, sep = " ")
TO <- unite(TO, "stop.time", stop.time, stop.time.ampm, sep = " ")

#converted AM/PM to military time for easy analysis
TO$stop.time<- (format(strptime(TO$stop.time, "%I:%M:%S %p"), "%H:%M:%S"))
FROM$start.time <- (format(strptime(FROM$start.time, "%I:%M:%S %p"), "%H:%M:%S"))
```

- Renamed & restructured the date columns as below. I separated month, day and year in order to make the analysis easier to group



```r
#separated the dates into day, month, year in FROM
library(tidyr)
FROM <- separate(FROM, start.date, c("month", "day", "year"), sep = "/")
View(FROM)

#separated the dates into day, month, year in TO
library(tidyr)
TO <- separate(TO, stop.date, c("month", "day", "year"), sep = "/")
View(TO)
```

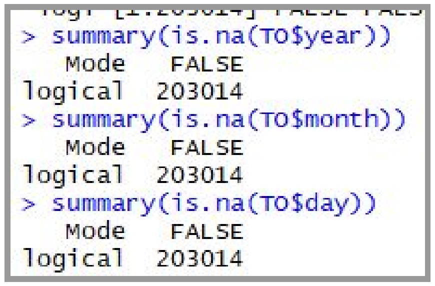
- Converted dates to numerics for easier grouping analysis



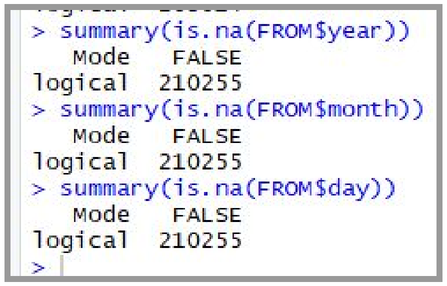
```
#converted dates to numeric for easier analysis
FROM$month <- as.numeric(unlist(FROM$month))
class(FROM$month)
FROM$day <- as.numeric(unlist(FROM$day))
class(FROM$day)
FROM$year <- as.numeric(unlist(FROM$year))
class(FROM$year)

TO$month <- as.numeric(unlist(TO$month))
class(TO$month)
TO$day <- as.numeric(unlist(TO$day))
class(TO$day)
TO$year <- as.numeric(unlist(TO$year))
class(TO$year)
```

2. Check for missing values

- Since the date columns are the most vital to my analysis I concentrated on this column to check for NA values as below in the TO and FROM datasets. A return of true would indicate an NA value. As per below there were no NA values found.

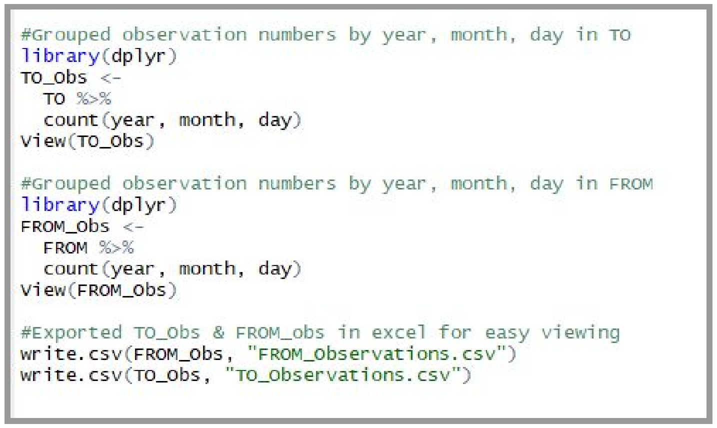


```
> summary(is.na(TO$year))
   Mode   FALSE
logical  203014
> summary(is.na(TO$month))
   Mode   FALSE
logical  203014
> summary(is.na(TO$day))
   Mode   FALSE
logical  203014
```



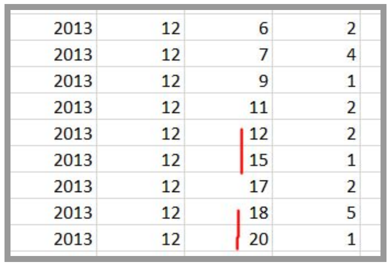
```
> summary(is.na(FROM$year))
   Mode   FALSE
logical  210255
> summary(is.na(FROM$month))
   Mode   FALSE
logical  210255
> summary(is.na(FROM$day))
   Mode   FALSE
logical  210255
>
```

- Grouped observations but the number of observations per day and exported them to excel for an easy analysis. The excels for TO and FROM may be found here.



```
#Grouped observation numbers by year, month, day in TO
library(dplyr)
TO_Obs <-
  TO %>%
  count(year, month, day)
View(TO_Obs)

#Grouped observation numbers by year, month, day in FROM
library(dplyr)
FROM_Obs <-
  FROM %>%
  count(year, month, day)
View(FROM_Obs)

#Exported TO_Obs & FROM_obs in excel for easy viewing
write.csv(FROM_Obs, "FROM_Observations.csv")
write.csv(TO_Obs, "TO_Observations.csv")
```

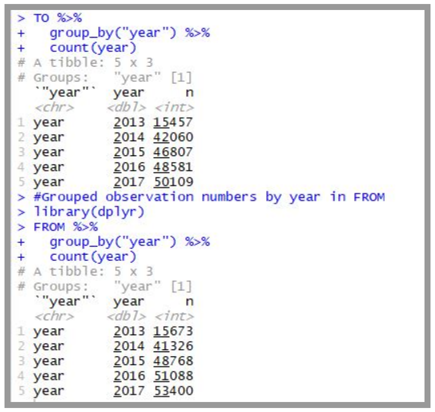
- Noticed within the excel that certain days were skipped on forecasting. However, since the total of the observations in the excel matches the total observations in the TO (203014) and FROM (210255) these missing dates were probably not caused by my data extraction, rather they were not included / observed in the original data set



2013	12	6	2
2013	12	7	4
2013	12	9	1
2013	12	11	2
2013	12	12	2
2013	12	15	1
2013	12	17	2
2013	12	18	5
2013	12	20	1

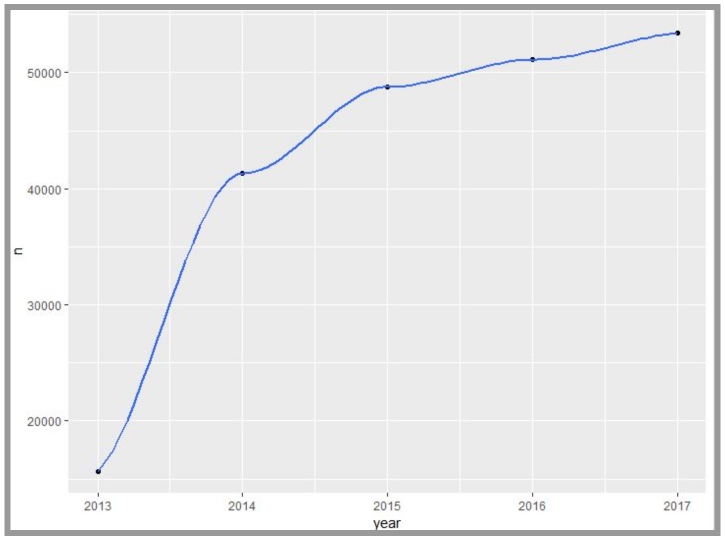
3. Overview on the number of observations per year, month & day

- First, I grouped the values and number of observations based on year

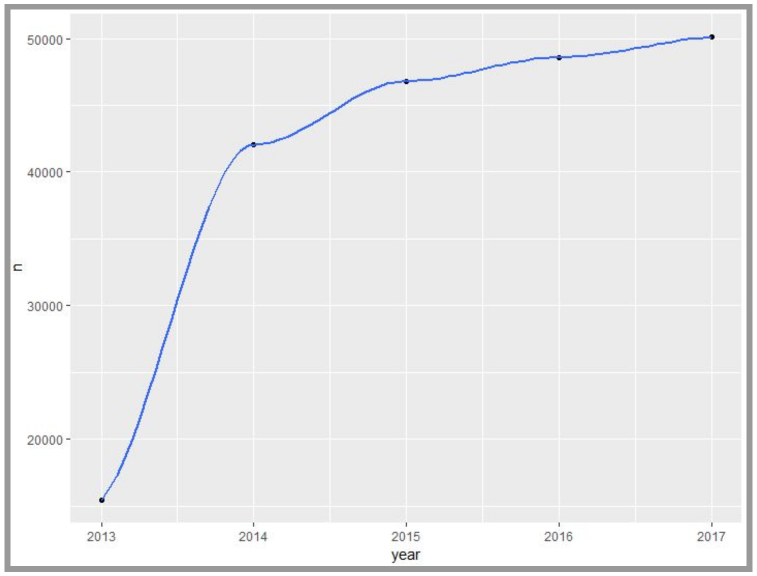


```
> TO %>%
+   group_by("year") %>%
+   count(year)
# A tibble: 5 x 3
# Groups:   "year" [1]
  `"year"`  year     n
  <chr>    <dbl> <int>
1 year      2013 15457
2 year      2014 42060
3 year      2015 46807
4 year      2016 48581
5 year      2017 50109
> #Grouped observation numbers by year in FROM
> library(dplyr)
> FROM %>%
+   group_by("year") %>%
+   count(year)
# A tibble: 5 x 3
# Groups:   "year" [1]
  `"year"`  year     n
  <chr>    <dbl> <int>
1 year      2013 15673
2 year      2014 41326
3 year      2015 48768
4 year      2016 51088
5 year      2017 53400
```

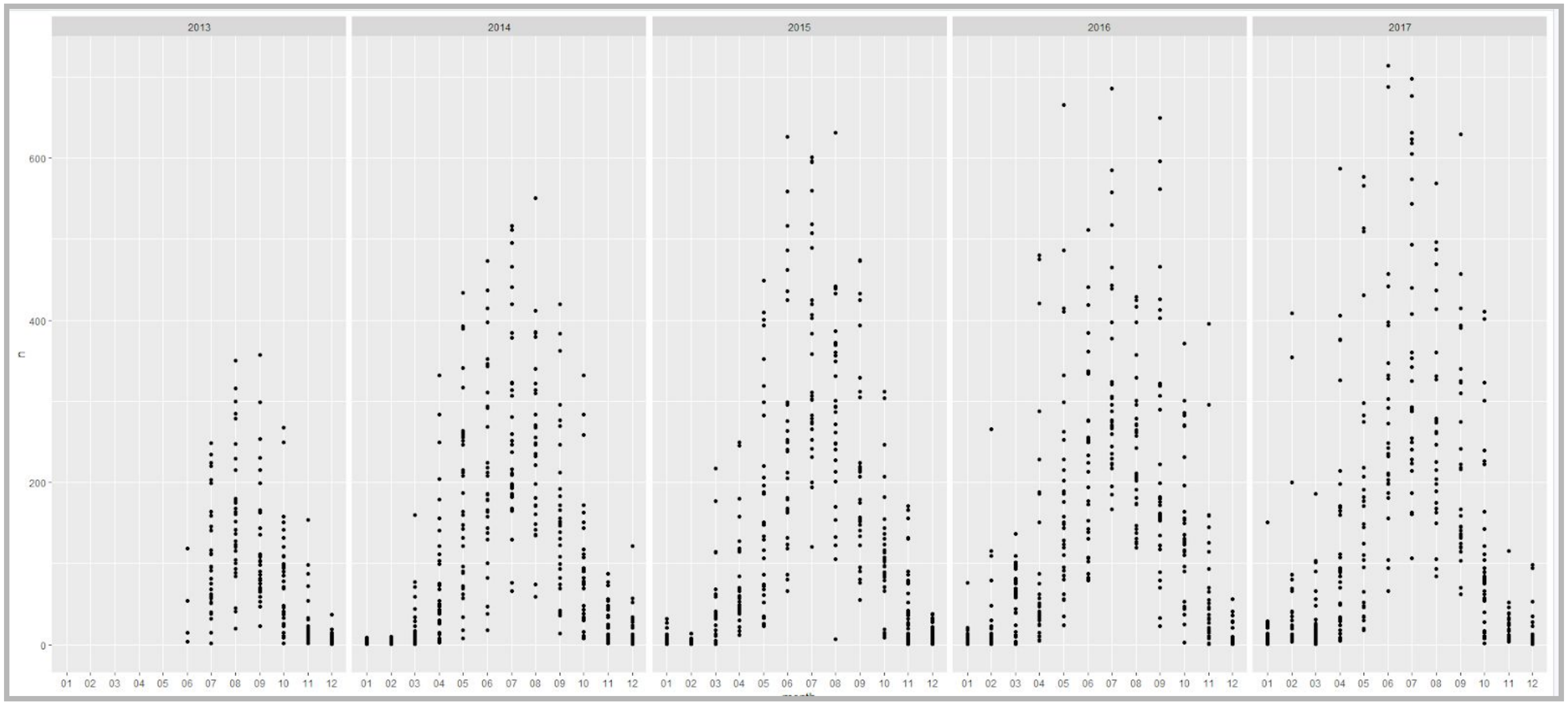
- visualized the number of observations per year in FROM



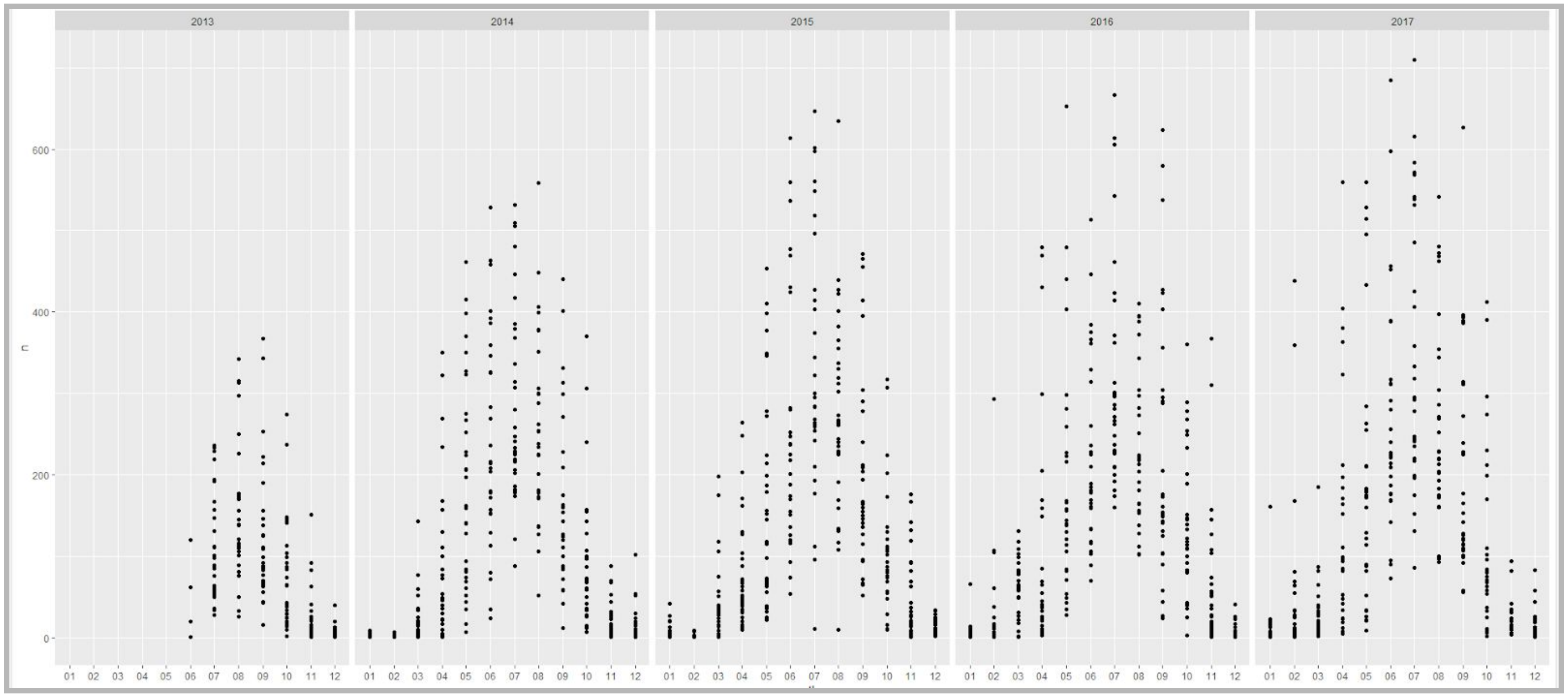
- visualized the number of observations per year in TO



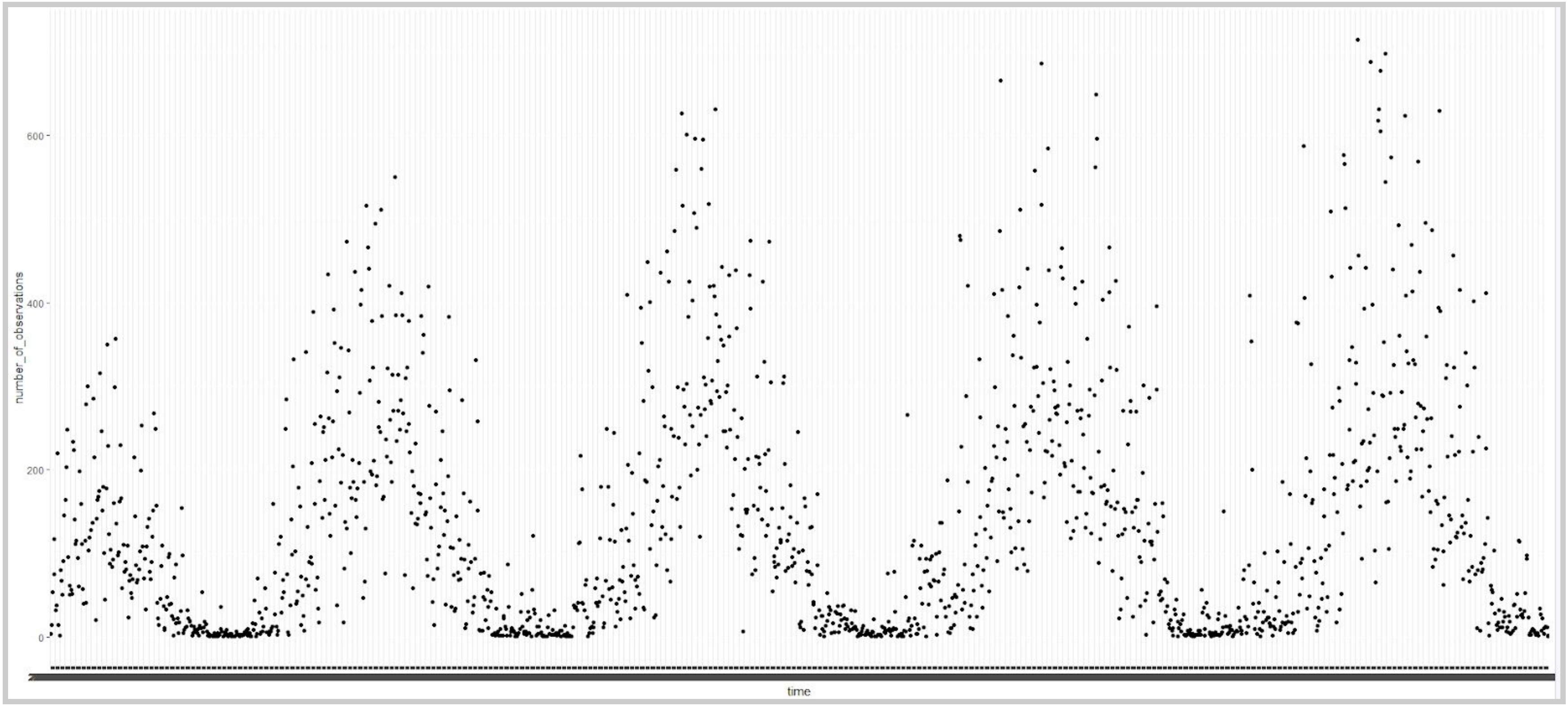
- FROM broken down by observation by month and year



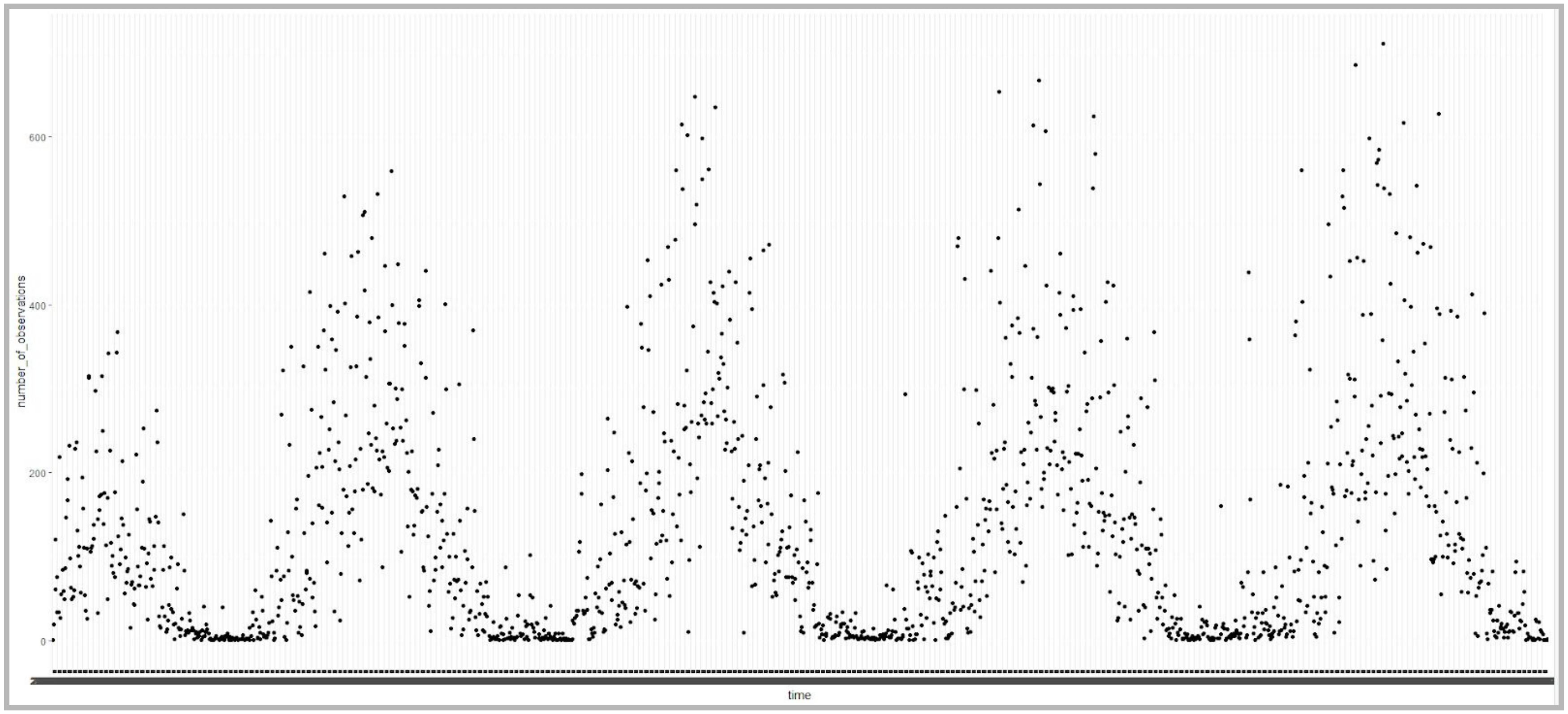
- TO broken down by observation by month and year



- In order to visualize the number of observations in FROM into days I created a new set with the year, day and month and n - a pattern of seasonality emerges



- Did the same thing with TO



Conclusion

After wrangling the data a few patterns emerged:

- Over the period of 2013 - 2017 the number of Divvy Bikes brought to and taken from the Divvy station increased - this makes sense as Divvy bikes began operations in 2013 so as brand awareness increased more people took advantage of using this service
- There is a clear pattern of seasonality as demand increases in the summer months and drops off during the winter month as seen in the final 4 visualizations