NYPD Shooting Incident

Jaeryang Baek

6/15/2021

Before starting

About data

In advance to starting any of the data analysis, It is required to download the `NYPD Shooting Incident Data` beforehand. You can download the data set by visiting the website https://catalog.data.gov/dataset and searching the term 'NYPD Shooting Incident Data (Historic)'. Make sure to have the csv file in the same folder as the Rmd file. I've renamed the csv file to `NYPD_Shooting_Incident_Data.csv` after downloading the file for the sake of convenience.

From 2006 to the end of the previous calendar year, every gunshot occurrence in New York City was documented in the data.

Areas of Focus

- When is the safest time to travel the city for a visitor?
- When is the safest/most dangerous season?
- Which boroughs should you avoid?
- Any correlation between the Incident count with the Murder count?

Importing libraries

Importing commonly used R libraries such as `tidyverse`, `lubridate`.

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

##
## Attaching package: 'lubridate'

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

Importing data

Importing the downloaded data from the local directory `./NYPD_Shooting_Incident_Data.csv`

```
file_dir <- "./NYPD_Shooting_Incident_Data.csv"

incidents <- read_csv(file_dir)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   INCIDENT_KEY = col_double(),
##   OCCUR_DATE = col_character(),
##   OCCUR_TIME = col_time(format = ""),
##   BORO = col_character(),
##   PRECINCT = col_double(),
##   JURISDICTION_CODE = col_double(),
##   LOCATION_DESC = col_character(),
##   STATISTICAL_MURDER_FLAG = col_logical(),
##   PERP_AGE_GROUP = col_character(),
##   PERP_SEX = col_character(),
##   PERP_RACE = col_character(),
##   VIC_AGE_GROUP = col_character(),
##   VIC_SEX = col_character(),
##   VIC_RACE = col_character(),
##   X_COORD_CD = col_number(),
##   Y_COORD_CD = col_number(),
##   Latitude = col_double(),
##   Longitude = col_double(),
##   Lon_Lat = col_character()
## )
```

or we could import the data from accessing the url directly from online

```
incidents <- read_csv("https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?accessType=DOWNLOAD")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   INCIDENT_KEY = col_double(),
##   OCCUR_DATE = col_character(),
##   OCCUR_TIME = col_time(format = ""),
##   BORO = col_character(),
##   PRECINCT = col_double(),
##   JURISDICTION_CODE = col_double(),
##   LOCATION_DESC = col_character(),
##   STATISTICAL_MURDER_FLAG = col_logical(),
##   PERP_AGE_GROUP = col_character(),
##   PERP_SEX = col_character(),
##   PERP_RACE = col_character(),
##   VIC_AGE_GROUP = col_character(),
##   VIC_SEX = col_character(),
##   VIC_RACE = col_character(),
##   X_COORD_CD = col_number(),
##   Y_COORD_CD = col_number(),
##   Latitude = col_double(),
##   Longitude = col_double(),
##   Lon_Lat = col_character()
## )
```

Tidy & Transform data

Tidying up and Transforming the data by removing the columns not necessary, as well as removing the rows with the missing data and filtering the data so that the obvious errors can be omitted as well.

After getting rid of the missing data, We change the format (variable types) of the columns such as date and time so that it's easier for us to manipulate the data for analysis and to work with in general.

```
incidents <- incidents %>%
  select(c(OCCUR_DATE,OCCUR_TIME, BORO, STATISTICAL_MURDER_FLAG))

incidents <- incidents[complete.cases(incidents),]

incidents <- mutate(incidents, OCCUR_DATE = mdy(OCCUR_DATE)) %>%
  mutate(DATETIME = ymd_hms(str_c(OCCUR_DATE, OCCUR_TIME))) %>%
  mutate(MURDER = STATISTICAL_MURDER_FLAG) %>%
  select(-c(OCCUR_DATE, OCCUR_TIME, STATISTICAL_MURDER_FLAG))

summary(incidents)
```

```
##      BORO              DATETIME                       MURDER
##  Length:23568       Min.   :2006-01-01 02:00:00   Mode :logical
##  Class :character   1st Qu.:2008-12-30 04:27:00   FALSE:19080
##  Mode  :character   Median :2012-02-26 03:35:00   TRUE :4488
##                     Mean   :2012-10-04 05:23:12
##                     3rd Qu.:2016-02-28 00:01:00
##                     Max.   :2020-12-31 23:45:00
```

```
incidents_by_boro <- incidents %>%
  group_by(BORO) %>% count()

incidents_by_boro_hour_dist <- incidents %>%
  mutate(TIME = hour(round_date(DATETIME, "hour"))) %>%
  select(c(BORO, TIME)) %>% table() %>%
  as.data.frame()

incidents_by_boro_month_dist <- incidents %>%
  mutate(MONTH = month(round_date(DATETIME, "month"))) %>%
  select(c(BORO, MONTH)) %>% table() %>%
  as.data.frame()

all_incidents_by_year <- incidents %>%
  mutate(YEAR = year(round_date(DATETIME, "year"))) %>%
  group_by(YEAR) %>% count()

colnames(all_incidents_by_year) <- c("YEAR", "INCIDENTS")
```

```r
all_murders_by_year <- incidents %>%
  filter(MURDER == TRUE) %>%
  mutate(YEAR = year(round_date(DATETIME, "year"))) %>%
  group_by(YEAR) %>% count()

colnames(all_murders_by_year) <- c("YEAR", "MURDERS")


by_year <- merge(all_incidents_by_year,all_murders_by_year,by="YEAR")
```

Analysis with Visualizations

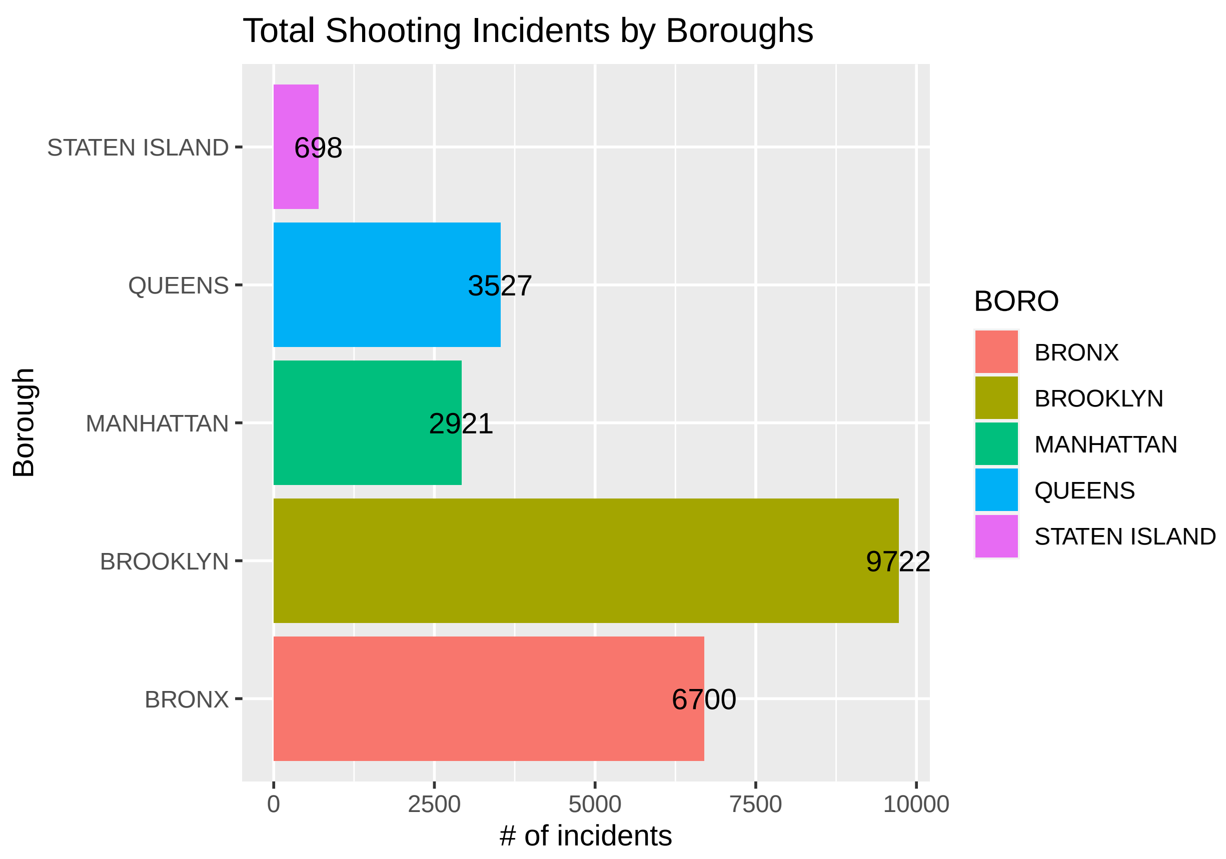
With the data obtained previously, We will do an analysis on when and where statistically is the most safest Boroughs in New York to travel.

For visualizations, we will utilize the data transformed above and give an additional analysis on the data as well.

Total shooting incidents by Boroughs?

First, we will take a look and visualize the data regarding the total occurrence of the shooting incidents classified by Boroughs.

```r
incidents_by_boro %>%
  ggplot(aes(BORO,n))+
  geom_col(aes(fill = BORO)) +
  coord_flip() +
  labs(title="Total Shooting Incidents by Boroughs",
       y="# of incidents", x= "Borough")+
  geom_text(aes(label = round(n, 1)), nudge_y= -3, color="black")
```



As you can observe from the plot above, `BROOKLYN` has the highest total shooting incidents and `STATEN ISLAND` has the lowest. From the data, we can deduce that if someone were to get into a shooting incident its most likely that the incident will occur in either `BROOKLYN` or `BRONX`. Therefore, I would suggest travelers to avoid those boroughs if possible, and if not, be on the lookout for any dangers.

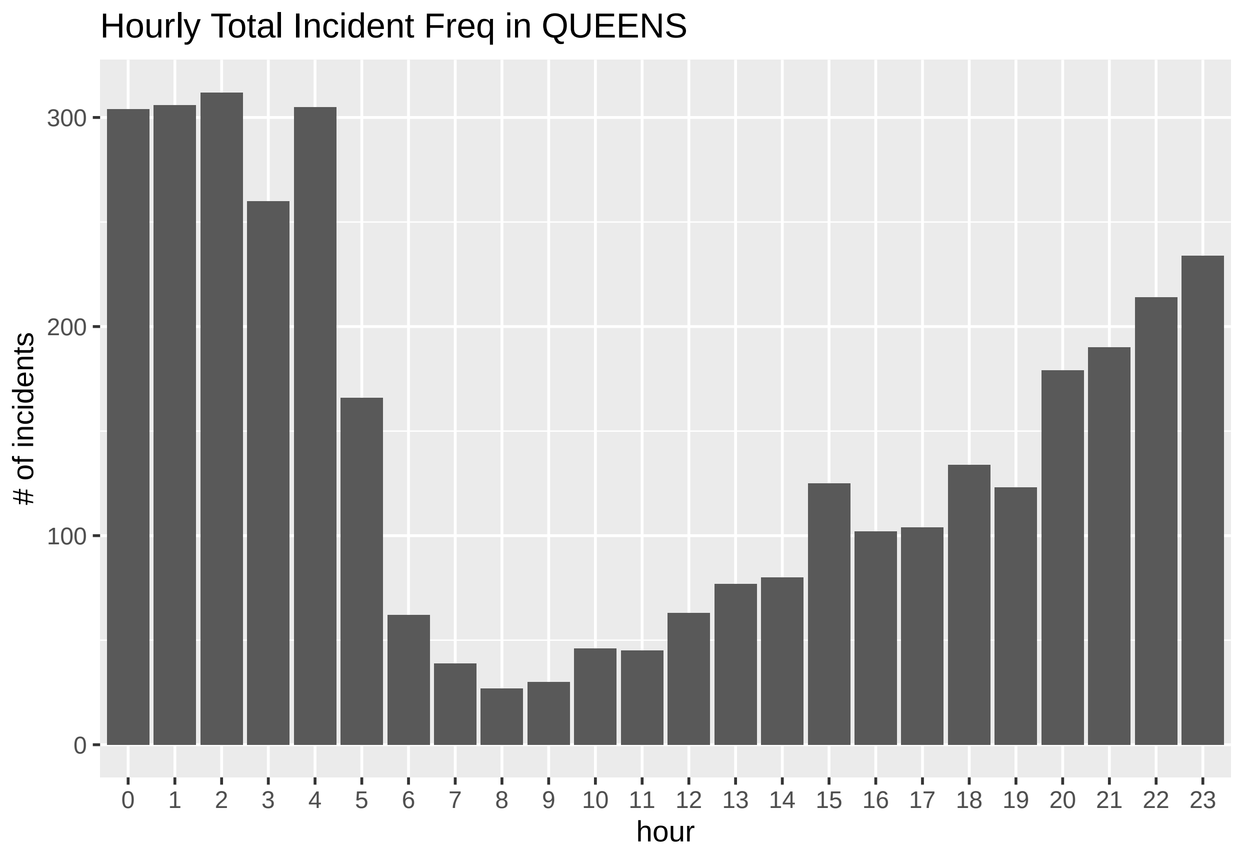
Total hourly shooting incidents by Boroughs?

Next up, we will take a look and create the bar plot regarding the total hourly occurrence of the shooting incidents classified by Boroughs.

For this specific visualization, we will choose the borough `QUEENS` to inspect further. Also, Feel free to change the variable `borough` to suit your needs.

```
borough <- "QUEENS"

incidents_by_boro_hour_dist %>% filter(BORO == borough) %>%
  ggplot(aes(TIME,Freq))+
  geom_col() +
  labs(title=str_c("Hourly Total Incident Freq in ",borough),
        y="# of incidents", x= "hour")
```

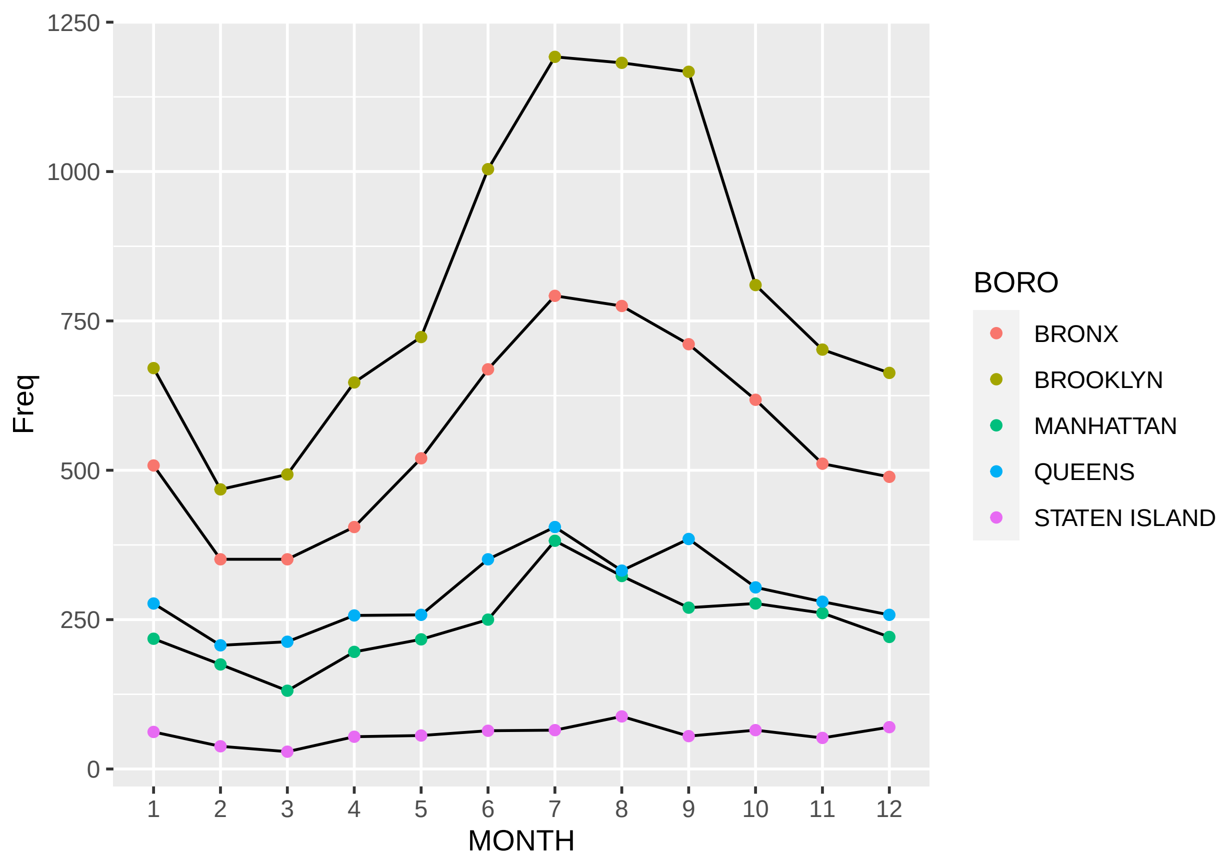


As you can see the incident occurrence spikes up during the night/early morning (0h - 4h) and drastically settles down during the morning/noon (6h - 12h). From this bar chart, we can infer that if a traveler were to avoid any danger, it would be wise to advise them to do not go around and minimize the movement in the borough during the night(0h - 4h) and try to visit places during the day(6h - 14h).

Total monthly shooting incidents of All Boroughs?

Last but not least, we will take a look and plot a line chart on the total monthly occurrence of the shooting incidents classified of All Boroughs at once.

```
incidents_by_boro_month_dist %>%
  ggplot(mapping = aes(x = MONTH, y = Freq)) +
  geom_line(aes(group = BORO)) +
  geom_point(aes(color = BORO))
```



As you can see from the line chart, when it comes to `STATEN ISLAND`, `QUEENS`, and `MANHATTAN` the incident rate stays pretty much the same throughout the year as oppose to `BRONX` and `BROOKLYN` peaking during the summer.

Why the incidents peaks during the summer is yet to be known because of the lack of data from the source.

From the visualization, we can conclude that its wise to be more cautious during the summer if in `BRONX` and `BROOKLYN`.

Modeling data

We model the data and explore the linear correlation between the `MURDERS` count with the `INCIDENTs` count by using function `lm()`.

```
mod <- lm(INCIDENTS ~ MURDERS, data = by_year)

summary(mod)
```

```
##
## Call:
## lm(formula = INCIDENTS ~ MURDERS, data = by_year)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -188.54  -88.61  -14.26   96.00  191.61
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 141.8987   115.9071   1.224    0.241
## MURDERS       4.7455     0.3977  11.932 1.01e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 125.8 on 14 degrees of freedom
## Multiple R-squared:  0.9105, Adjusted R-squared:  0.9041
## F-statistic: 142.4 on 1 and 14 DF,  p-value: 1.007e-08
```

```
by_year %>% slice_min(INCIDENTS)
```

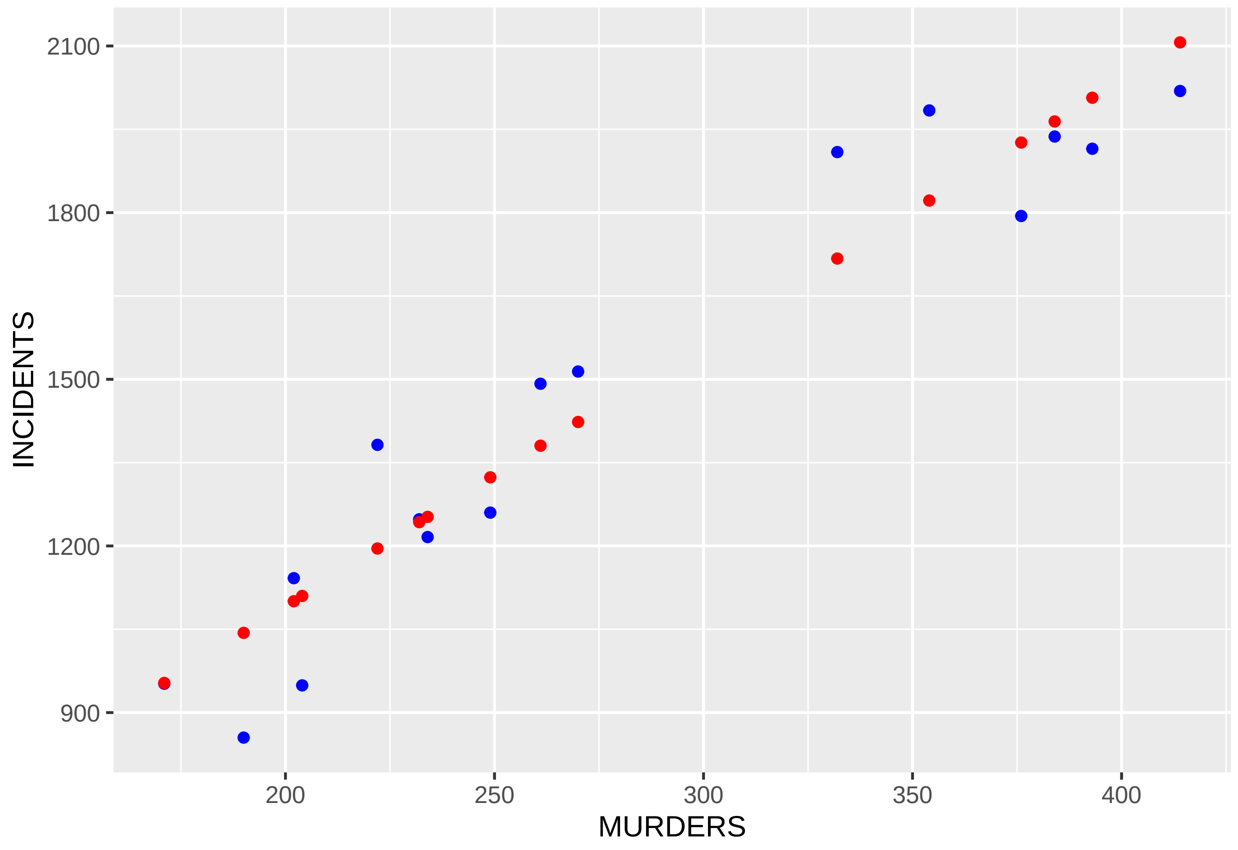
```
##   YEAR INCIDENTS MURDERS
## 1 2006       855     190
```

```
by_year %>% slice_max(INCIDENTS)
```

```
##   YEAR INCIDENTS MURDERS
## 1 2007      2019     414
```

```
by_year_w_pred <- by_year %>%
  mutate(pred = predict(mod))


by_year_w_pred %>% ggplot() +
  geom_point(aes(x = MURDERS, y = INCIDENTS), color = "blue") +
  geom_point(aes(x = MURDERS, y = pred), color = "red")
```



From the plot above, we can clearly identify the linear correlation.

Conclusion & Identifying Bias

In conclusion, if you know any friends and families whom are planning on travelling to New York in the near future, make sure to let them know that

- It can be particularly dangerous during the night.
- Be more cautious during the summer in `BRONX` and `BROOKLYN`.
- Avoid `BROOKLYN` or `BRONX` if possible.

As I do not have any personal bias against any boroughs in New York, I doubt there's been any skewed analysis for that specific reason. But the data that's been used for this analysis has not been adjusted to the total population (is not per capita) nor to the area size of the borough, which are crucial factors when it comes to understanding the whole picture. Because of this reason, we cannot be 100% certain that `BROOKLYN` or `BRONX` is the most dangerous part of the town statistically.

Session Info

```
sessionInfo()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 21390)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
## system code page: 65001
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.7.10 forcats_0.5.1    stringr_1.4.0    dplyr_1.0.6
##  [5] purrr_0.3.4      readr_1.4.0      tidyr_1.1.3      tibble_3.1.2
##  [9] ggplot2_3.3.3    tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.1.1  xfun_0.23         haven_2.4.1       colorspace_2.0-1
##  [5] vctrs_0.3.8       generics_0.1.0    htmltools_0.5.1.1 yaml_2.2.1
##  [9] utf8_1.2.1        rlang_0.4.11      pillar_1.6.1      glue_1.4.2
## [13] withr_2.4.2       DBI_1.1.1         dbplyr_2.1.1      modelr_0.1.8
## [17] readxl_1.3.1      lifecycle_1.0.0   munsell_0.5.0     gtable_0.3.0
## [21] cellranger_1.1.0  rvest_1.0.0       evaluate_0.14     labeling_0.4.2
## [25] knitr_1.33        curl_4.3.1        fansi_0.5.0       highr_0.9
## [29] broom_0.7.7       Rcpp_1.0.6        scales_1.1.1      backports_1.2.1
## [33] jsonlite_1.7.2    farver_2.1.0      fs_1.5.0          hms_1.1.0
## [37] digest_0.6.27     stringi_1.6.1     grid_4.1.0        cli_2.5.0
## [41] tools_4.1.0       magrittr_2.0.1    crayon_1.4.1      pkgconfig_2.0.3
## [45] ellipsis_0.3.2    xml2_1.3.2        reprex_2.0.0      assertthat_0.2.1
## [49] rmarkdown_2.8     httr_1.4.2        rstudioapi_0.13   R6_2.5.0
## [53] compiler_4.1.0
```