NewYork_Incidents_Project1

Introduction

I am going to be walking through the steps provided in Week 3 of the Data Science as a Field course. The assignment is to reproduce a few steps in which data scientists gather data from reputable sources, clean that data and transform for them to use during their analysis, and finally to run analysis and identify possible biases during their work. This was a fun assignment and I took a few approaches to get the most out of it.

First: Read the data in and transform the data

I started by inputting the data into a variable `NYPD_data`.

```
library(readr)
```

```
## Warning: package 'readr' 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(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

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

```
## v ggplot2 3.3.3     v dplyr   1.0.6
## v tibble  3.1.0     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v purrr   0.3.4
```

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

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

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

## Warning: package 'forcats' was built under R version 4.0.5

## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

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

```
##
## -- 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()
## )
```

This is the data before the cleanup, a summary gives you an idea of the current values, and the datatypes.

```
summary(NYPD_data)
```

```
##   INCIDENT_KEY        OCCUR_DATE         OCCUR_TIME           BORO
##  Min.   :  9953245   Length:23568       Length:23568       Length:23568
##  1st Qu.: 55317014   Class :character   Class1:hms         Class :character
##  Median : 83365370   Mode  :character   Class2:difftime    Mode  :character
##  Mean   :102218616                      Mode  :numeric
##  3rd Qu.:150772442
##  Max.   :222473262
##
```

```
##     PRECINCT      JURISDICTION_CODE LOCATION_DESC      STATISTICAL_MURDER_FLAG
##  Min.   :  1.00   Min.   :0.0000    Length:23568       Mode :logical
##  1st Qu.: 44.00   1st Qu.:0.0000    Class :character   FALSE:19080
##  Median : 69.00   Median :0.0000    Mode  :character   TRUE :4488
##  Mean   : 66.21   Mean   :0.3323
##  3rd Qu.: 81.00   3rd Qu.:0.0000
##  Max.   :123.00   Max.   :2.0000
##                   NA's   :2
##  PERP_AGE_GROUP       PERP_SEX          PERP_RACE         VIC_AGE_GROUP
##  Length:23568       Length:23568       Length:23568       Length:23568
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    VIC_SEX            VIC_RACE           X_COORD_CD        Y_COORD_CD
##  Length:23568       Length:23568       Min.   : 914928   Min.   :125757
##  Class :character   Class :character   1st Qu.: 999900   1st Qu.:182565
##  Mode  :character   Mode  :character   Median :1007645   Median :193482
##                                        Mean   :1009363   Mean   :207312
##                                        3rd Qu.:1016807   3rd Qu.:239163
##                                        Max.   :1066815   Max.   :271128
##
##     Latitude       Longitude        Lon_Lat
##  Min.   :40.51   Min.   :-74.25   Length:23568
##  1st Qu.:40.67   1st Qu.:-73.94   Class :character
##  Median :40.70   Median :-73.92   Mode  :character
##  Mean   :40.74   Mean   :-73.91
##  3rd Qu.:40.82   3rd Qu.:-73.88
##  Max.   :40.91   Max.   :-73.70
##
```

Data cleanup

I cleaned up the data in a few steps. First I changed the data type of the occurrence date. I also could have changed the variable need for ease of use, but chose not to during this assignment.

```
NYPD_data$OCCUR_DATE <- mdy(NYPD_data$OCCUR_DATE)
summary(NYPD_data)
```

```
##   INCIDENT_KEY         OCCUR_DATE          OCCUR_TIME           BORO
##  Min.   :  9953245   Min.   :2006-01-01   Length:23568      Length:23568
##  1st Qu.: 55317014   1st Qu.:2008-12-30   Class1:hms        Class :character
##  Median : 83365370   Median :2012-02-26   Class2:difftime   Mode  :character
##  Mean   :102218616   Mean   :2012-10-03   Mode  :numeric
##  3rd Qu.:150772442   3rd Qu.:2016-02-28
##  Max.   :222473262   Max.   :2020-12-31
##
##     PRECINCT      JURISDICTION_CODE LOCATION_DESC      STATISTICAL_MURDER_FLAG
##  Min.   :  1.00   Min.   :0.0000    Length:23568       Mode :logical
##  1st Qu.: 44.00   1st Qu.:0.0000    Class :character   FALSE:19080
##  Median : 69.00   Median :0.0000    Mode  :character   TRUE :4488
```

```
##  Mean   : 66.21   Mean   :0.3323
##  3rd Qu.: 81.00   3rd Qu.:0.0000
##  Max.   :123.00   Max.   :2.0000
##                   NA's   :2
##  PERP_AGE_GROUP       PERP_SEX          PERP_RACE         VIC_AGE_GROUP
##  Length:23568       Length:23568       Length:23568       Length:23568
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    VIC_SEX            VIC_RACE           X_COORD_CD        Y_COORD_CD
##  Length:23568       Length:23568       Min.   : 914928   Min.   :125757
##  Class :character   Class :character   1st Qu.: 999900   1st Qu.:182565
##  Mode  :character   Mode  :character   Median :1007645   Median :193482
##                                        Mean   :1009363   Mean   :207312
##                                        3rd Qu.:1016807   3rd Qu.:239163
##                                        Max.   :1066815   Max.   :271128
##
##     Latitude       Longitude        Lon_Lat
##  Min.   :40.51   Min.   :-74.25   Length:23568
##  1st Qu.:40.67   1st Qu.:-73.94   Class :character
##  Median :40.70   Median :-73.92   Mode  :character
##  Mean   :40.74   Mean   :-73.91
##  3rd Qu.:40.82   3rd Qu.:-73.88
##  Max.   :40.91   Max.   :-73.70
##
```

I removed a few columns to keep the necessary information on hand. There are a few other columns I could have removed but chose to remove them at the grouping phase, as analysis could have been run on things like: Age groups, race, and sex of the perp and victim.

```
NYPD_data <- NYPD_data %>% select(-c(Latitude, Longitude, X_COORD_CD, Y_COORD_CD, Lon_Lat))
```

Second: Start grouping the data for analysis

I grouped the data in multiple ways to have some insight into the effects of location, and time of year. ### By Borough and Precinct First we need to group it based on the geographical location of the incidents. Based on the data provided, I grouped them based on the borough it occurred, and the precinct that would have reacted to it.

```
NYPD_data_grouped_byPRECANDBORO <- NYPD_data %>% group_by(BORO, PRECINCT) %>%
  summarise(count=n()) %>% select(everything()) %>% ungroup()
```

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

```
NYPD_data_grouped_byPRECANDBORO
```

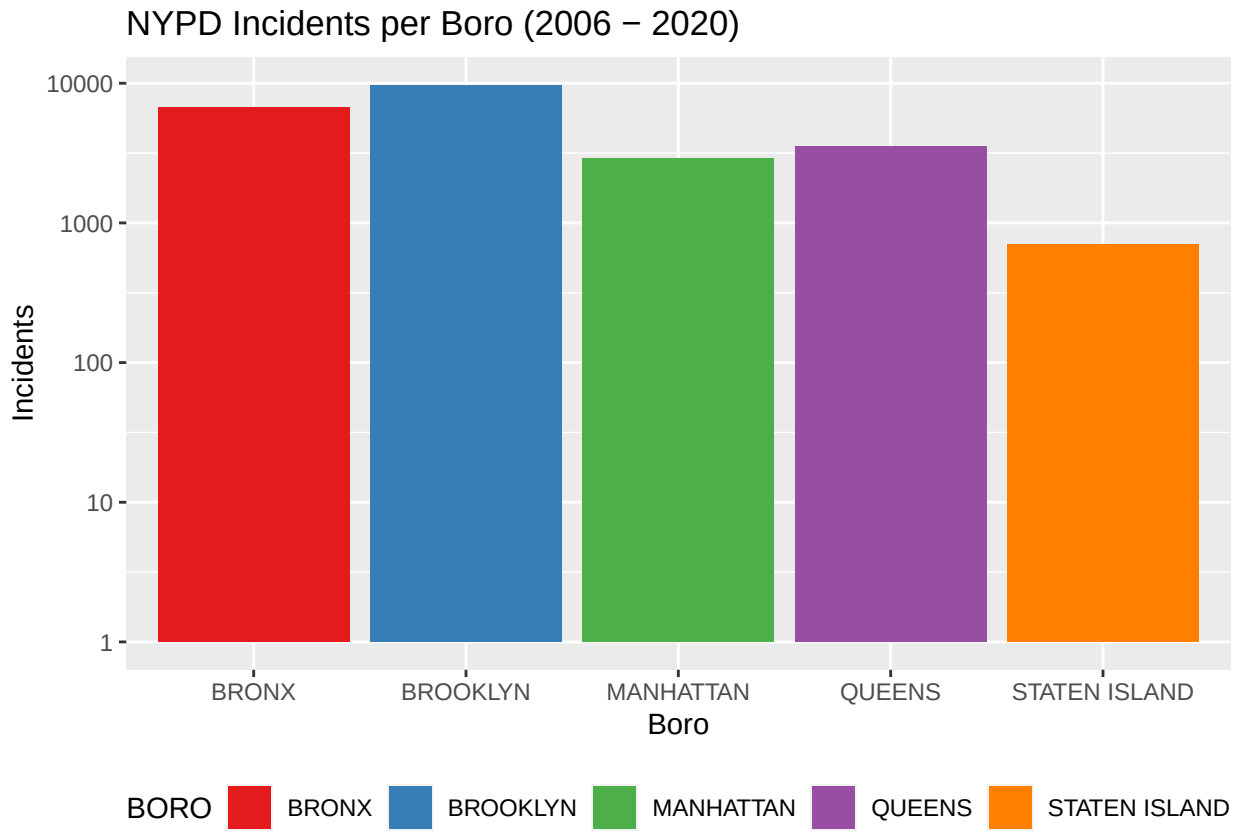
```
## # A tibble: 77 x 3
##    BORO  PRECINCT count
```

```
##    <chr>    <dbl> <int>
##  1 BRONX       40   759
##  2 BRONX       41   417
##  3 BRONX       42   725
##  4 BRONX       43   657
##  5 BRONX       44   842
##  6 BRONX       45   148
##  7 BRONX       46   779
##  8 BRONX       47   815
##  9 BRONX       48   639
## 10 BRONX       49   296
## # ... with 67 more rows
```

For the visualization, I chose to exclude the precinct in the chart as it would have added unnecessary clutter. Based on the data provided, below is a visualization of all incidents in the years 2006-2020 put into a bar chart based on the borough.

```
NYPD_data_grouped_byBORO <- NYPD_data %>% group_by(BORO) %>%
  summarise(total=n()) %>% select(everything()) %>% ungroup()

NYPD_data_grouped_byBORO %>% ggplot(aes(factor(BORO), y=total, fill=BORO)) + geom_bar(stat="identity") +
```



Based on the visuals, there doesn't seem to be a connection between location and incidents, but there are some boroughs that have less crime compared to others. We can also break it down into precincts to see which exact neighborhoods provide the best and worst safety for its inhabitants.

By date

Breaking down the data and viewing it based on the date it occurred will help us see the increase of crime over time. I first started by grouping it based on the date: `dd-mm-yy`

```
NYPD_data_grouped_bydate <- NYPD_data %>% group_by(OCCUR_DATE) %>% summarise(total=n()) %>% select(every
NYPD_data_grouped_bydate
```

```
## # A tibble: 5,054 x 2
##    OCCUR_DATE total
##    <date>     <int>
##  1 2006-01-01     8
##  2 2006-01-02     4
##  3 2006-01-03     4
##  4 2006-01-04     4
##  5 2006-01-05     4
##  6 2006-01-06     4
##  7 2006-01-07     2
##  8 2006-01-08     4
##  9 2006-01-09     9
## 10 2006-01-10     5
## # ... with 5,044 more rows
```

But that gives us too many rows, so if we want to make it by month we do:

```
NYPD_data_grouped_monthly <- NYPD_data %>%
  mutate(month = month(OCCUR_DATE, label=TRUE), year= year(OCCUR_DATE)) %>% group_by(month, year) %>%
  summarise(total=n()) %>% select(everything()) %>% ungroup() %>% arrange(year)
```

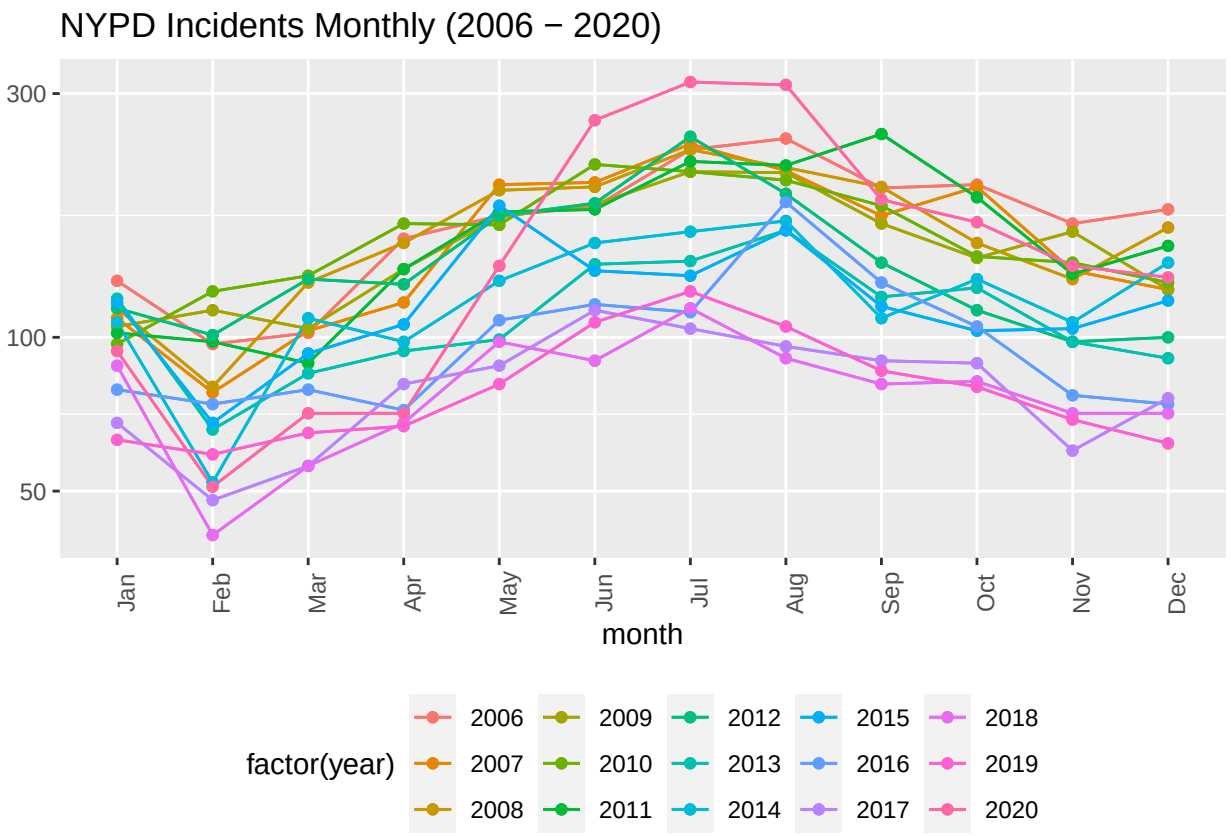
```
## `summarise()` has grouped output by 'month'. You can override using the `.groups` argument.
```

```
NYPD_data_grouped_monthly
```

```
## # A tibble: 180 x 3
##    month  year total
##    <ord> <dbl> <int>
##  1 Jan    2006   129
##  2 Feb    2006    97
##  3 Mar    2006   102
##  4 Apr    2006   156
##  5 May    2006   173
##  6 Jun    2006   180
##  7 Jul    2006   233
##  8 Aug    2006   245
##  9 Sep    2006   196
## 10 Oct    2006   199
## # ... with 170 more rows
```

This provides us with the below chart, which shows the years 2006-2020 all on the same chart, and shows that the incident rates have been fluctuating over time with the peak being in July of 2020.

```
NYPD_data_grouped_monthly %>% ggplot(aes(x = month, y=total, group=factor(year))) + geom_line(aes(color=
```



To get a yearly overview I then grouped them based on the year of the incident.

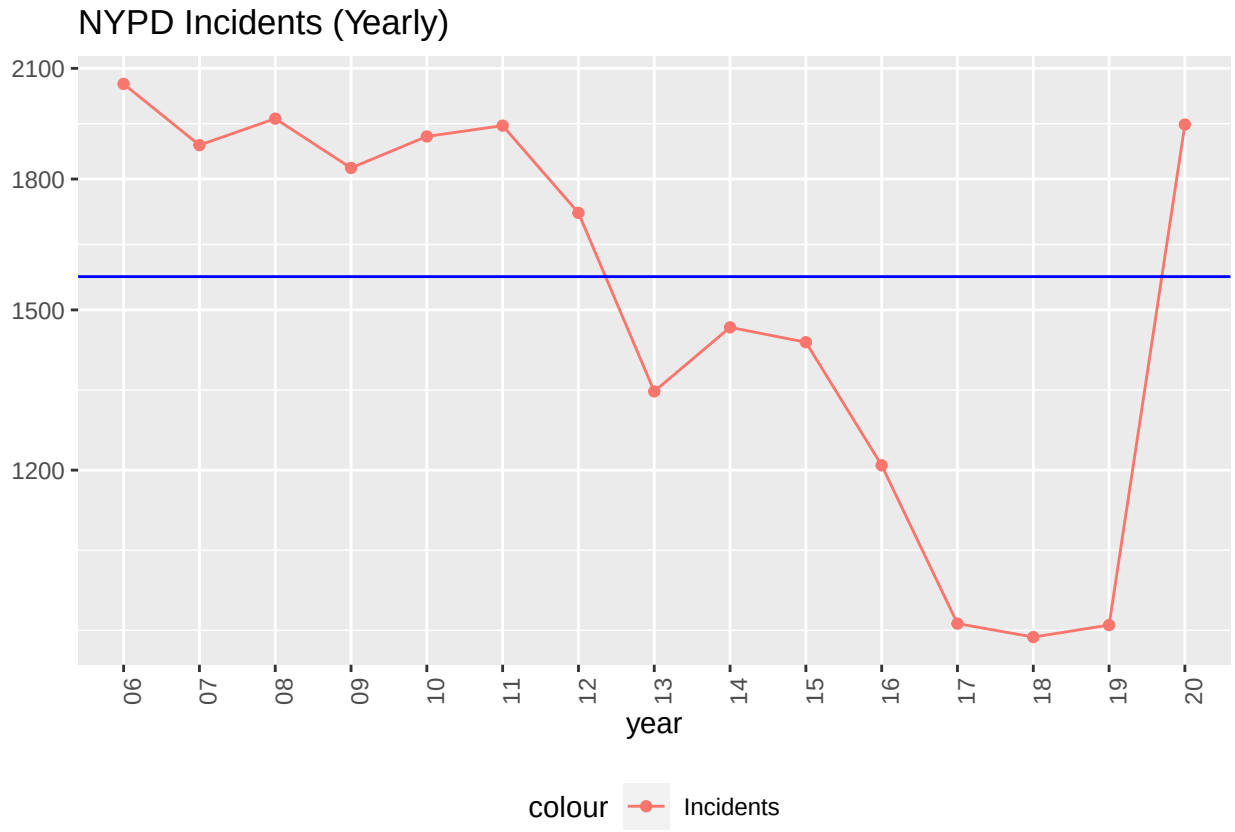
```
NYPD_data_grouped_yearly <- NYPD_data %>%
  mutate(year= format(OCCUR_DATE, "%y")) %>% group_by(year) %>%
  summarise(total=n()) %>% select(everything()) %>% ungroup()
NYPD_data_grouped_yearly
```

```
## # A tibble: 15 x 2
##    year  total
##    <chr> <int>
##  1 06     2055
##  2 07     1887
##  3 08     1958
##  4 09     1828
##  5 10     1910
##  6 11     1939
##  7 12     1717
##  8 13     1339
##  9 14     1464
## 10 15     1434
## 11 16     1208
## 12 17      969
## 13 18      951
```

```
## 14 19       967
## 15 20      1942
```

With the following chart as the result

```
NYPD_data_grouped_yearly %>% ggplot(aes(x = year, y=total, group=1)) +
geom_line(aes(color="Incidents")) + geom_point(aes(color="Incidents")) + scale_y_log10() + theme(legend
```



The blue line is an average of the yearly totals, and shows that between 2006 and 2012 the number of incidents were above the average over 14 years. We can also see that there is a steep rise in incidents in the year 2020, but there could be a correlation between the introduction of stay-at-home orders and the rise in incidents.

Analysis

To go deeper into the data, we can import another dataset provided by https://data.gov. This dataset has the population amount for the City of New York based on the Borough, and it can be used to analyze the data further.

```
nyc_pop <- read_csv("https://data.cityofnewyork.us/api/views/97pn-acdf/rows.csv?accessType=DOWNLOAD")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Borough = col_character(),
```

```
##   Age = col_character(),
##   `2010` = col_double(),
##   `2015` = col_double(),
##   `2020` = col_double(),
##   `2025` = col_double(),
##   `2030` = col_double(),
##   `2035` = col_double(),
##   `2040` = col_double()
## )
```

```
nyc_pop
```

```
## # A tibble: 114 x 9
##    Borough   Age   `2010` `2015` `2020` `2025` `2030` `2035` `2040`
##    <chr>     <chr>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 NYC Total 0-4   521990 535209 545778 547336 542426 540523 546426
##  2 NYC Total 15-19 539844 505783 492532 519298 535024 546062 546750
##  3 NYC Total 20-24 647483 646075 606203 591683 625253 643728 657403
##  4 NYC Total 25-29 736105 770396 763956 715824 698195 740437 762757
##  5 NYC Total 30-34 667657 707726 743916 740268 693684 675497 715486
##  6 NYC Total 35-39 592299 611239 649594 684249 682964 639237 621899
##  7 NYC Total 40-44 571825 550097 569628 606185 638148 637517 596493
##  8 NYC Total 45-49 570273 535998 517668 537516 571723 600792 600514
##  9 NYC Total 50-54 546204 552074 520597 504322 523815 556586 584164
## 10 NYC Total 55-59 479661 493997 501239 474319 459574 477052 506390
## # ... with 104 more rows
```

The dataset is broken down into ages so we can create first group the dataset to make it easier to work with.

```
NYC_pop_grouped <- nyc_pop %>% group_by(Borough) %>% summarise() %>% ungroup()
NYC_pop_grouped
```

```
## # A tibble: 6 x 1
##   Borough
##   <chr>
## 1 Bronx
## 2 Brooklyn
## 3 Manhattan
## 4 NYC Total
## 5 Queens
## 6 Staten Island
```

```
agg_2010 <- aggregate(x = nyc_pop$"2010", by = list(nyc_pop$Borough), FUN=sum)
agg_2010
```

```
##         Group.1        x
## 1         Bronx  2770216
## 2      Brooklyn  5105822
## 3     Manhattan  3171746
## 4     NYC Total 16485248
## 5        Queens  4500004
## 6 Staten Island   937460
```

```
agg_2010 <- agg_2010 %>% rename(Borough = Group.1, "2010" = x)

agg_2015 <- aggregate(x = nyc_pop$"2015", by = list(nyc_pop$Borough), FUN=sum)
agg_2015
```

```
##         Group.1        x
## 1         Bronx  2831450
## 2      Brooklyn  5205688
## 3     Manhattan  3221394
## 4     NYC Total 16794228
## 5        Queens  4578978
## 6 Staten Island   956718
```

```
agg_2015 <- agg_2015 %>% rename(Borough = Group.1, "2015" = x)

agg_2020 <- aggregate(x = nyc_pop$"2020", by = list(nyc_pop$Borough), FUN=sum)
agg_2020
```

```
##         Group.1        x
## 1         Bronx  2893576
## 2      Brooklyn  5296904
## 3     Manhattan  3276564
## 4     NYC Total 17101944
## 5        Queens  4660590
## 6 Staten Island   974310
```

```
agg_2020 <- agg_2020 %>% rename(Borough = Group.1, "2020" = x)

nyc_pop_grouped <- merge( x = agg_2010, y=NYC_pop_grouped, all.Borough=True)
nyc_pop_grouped <- merge( x = agg_2015, y=nyc_pop_grouped, all.Borough=True)
nyc_pop_grouped <- merge( x = agg_2020, y=nyc_pop_grouped, all.Borough=True)

nyc_pop_grouped <- nyc_pop_grouped[- grep("NYC Total", nyc_pop_grouped$Borough),]
nyc_pop_grouped
```

```
##         Borough    2020    2015    2010
## 1         Bronx 2893576 2831450 2770216
## 2      Brooklyn 5296904 5205688 5105822
## 3     Manhattan 3276564 3221394 3171746
## 5        Queens 4660590 4578978 4500004
## 6 Staten Island  974310  956718  937460
```

```
Bronxpop <- nyc_pop_grouped %>% filter(Borough == "Bronx")
Bronxpop
```

```
##   Borough    2020    2015    2010
## 1   Bronx 2893576 2831450 2770216
```

```
Bronxpop <- Bronxpop[1,2]
Bronxpop
```

```
## [1] 2893576
```

```
BronxInc <- NYPD_data_grouped_byBORO %>% filter(BORO == "BRONX")
BronxInc <- as.vector(BronxInc$total[1])
BronxInc
```

```
## [1] 6700
```

```
mod <- lm(BronxInc ~ Bronxpop)
summary(mod)
```

```
##
## Call:
## lm(formula = BronxInc ~ Bronxpop)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     6700         NA      NA       NA
## Bronxpop          NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
```

I attempted at making a model here but unfortunately it did not work. I needed to combine the population data frame into the NYPD incident data frame. Then make a model that attempts to draw a connection between population density and the number of incidents in that borough. Unfortunately, this was not explained enough at this stage of the program for me to go deeper into my development. I hope that this does not hurt my grades.

Conclusion

Further analysis can be done when it relates to the data of NYC to pinpoint the exact locations of the incidents, create a heatmap of the city based on the occurrences and provide better insight into the number of incidents related to the number of population in each area. I believe that this has correlation between the two, but that is based on my biases.

I believe that the bias that the data may have is based on the reporting of the incidents. Based on my analysis, the biases that may have occurred are: * The push to find a correlation between area and number of incidents, despite there being no obvious correlation in the data * How I grouped the data, as my need to avoid a clutter in the graph led me to make it based on borough and not on precinct. We may see more information when we break it down further than I have, and my personal bias has stopped me from going deeper into detail * and, as mentioned above, the larger bubble of bias is outside of my power as a data analyst. The data that is provided to me from sources such as https://data.gov is reported by local precincts, and that has an effect on the collection of the data as it requires each and every member of those precincts to provide that data without their personal bias affecting them.

Overall, this assignment gives great insight into the process of reproducing analysis and insight into how to do so. Peer reviewed work will help to mitigate the interference of personal bias in the workplace, and on data that may have an effect on other peoples actions.