NYPD Shooting Incident Data Report

2023-12-12

This project focuses on tidying, cleaning, organizing, visualizing, and analyzing the data from the NYPD Shooting Data Report (Historical). It is completely accessible to the public and can be found at https://catalog.data.gov/dataset/nypd-shooting-incident-data-historic.

Import Libraries and Data

The first step of importing the necessary libraries is essential to enable working with the data from the dataset of interest. In this case, the code is reading in the NYPD Shooting Data report as a CSV file.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(dplyr)
library(ggplot2)
library(RColorBrewer)
url_in <- "https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?accessType=DOWNLOAD"
urls <- str_c(url_in)
urls
```

```
## [1] "https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?accessType=DOWNLOAD"
```

```
NYPD_data = read.csv(urls[1])

#Summary of the Data
summary(NYPD_data)
```

```
##   INCIDENT_KEY        OCCUR_DATE         OCCUR_TIME            BORO
##  Min.   :  9953245   Length:27312       Length:27312       Length:27312
##  1st Qu.: 63860880   Class :character   Class :character   Class :character
##  Median : 90372218   Mode  :character   Mode  :character   Mode  :character
##  Mean   :120860536
##  3rd Qu.:188810230
```

```
##  Max.   :261190187
##
##  LOC_OF_OCCUR_DESC     PRECINCT      JURISDICTION_CODE LOC_CLASSFCTN_DESC
##  Length:27312       Min.   :  1.00   Min.   :0.0000    Length:27312
##  Class :character   1st Qu.: 44.00   1st Qu.:0.0000    Class :character
##  Mode  :character   Median : 68.00   Median :0.0000    Mode  :character
##                     Mean   : 65.64   Mean   :0.3269
##                     3rd Qu.: 81.00   3rd Qu.:0.0000
##                     Max.   :123.00   Max.   :2.0000
##                                      NA's   :2
##  LOCATION_DESC      STATISTICAL_MURDER_FLAG PERP_AGE_GROUP
##  Length:27312       Length:27312            Length:27312
##  Class :character   Class :character        Class :character
##  Mode  :character   Mode  :character        Mode  :character
##
##
##
##
##    PERP_SEX           PERP_RACE         VIC_AGE_GROUP        VIC_SEX
##  Length:27312       Length:27312       Length:27312       Length:27312
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    VIC_RACE            X_COORD_CD        Y_COORD_CD        Latitude
##  Length:27312       Min.   : 914928   Min.   :125757   Min.   :40.51
##  Class :character   1st Qu.:1000029   1st Qu.:182834   1st Qu.:40.67
##  Mode  :character   Median :1007731   Median :194487   Median :40.70
##                     Mean   :1009449   Mean   :208127   Mean   :40.74
##                     3rd Qu.:1016838   3rd Qu.:239518   3rd Qu.:40.82
##                     Max.   :1066815   Max.   :271128   Max.   :40.91
##                                                        NA's   :10
##    Longitude         Lon_Lat
##  Min.   :-74.25   Length:27312
##  1st Qu.:-73.94   Class :character
##  Median :-73.92   Mode  :character
##  Mean   :-73.91
##  3rd Qu.:-73.88
##  Max.   :-73.70
##  NA's   :10
```

Ensure the data types are correct:

After checking the summary and key aspects of the data, it is important to go through and check each object and correct the way each variable is being viwed in the code. For example, ensuring the "DATE" column is being treated as a date allows for accurate data manipulation later on.

```
#next make sure each column/variable is the correct type
NYPD_data <- NYPD_data %>%
  mutate(INCIDENT_KEY = as.character(INCIDENT_KEY))
NYPD_data <- NYPD_data %>%
  mutate(OCCUR_DATE = as.Date(OCCUR_DATE, format="%m/%d/%y"))
```

```r
NYPD_data <- NYPD_data %>%
  mutate(OCCUR_TIME = as.POSIXct(OCCUR_TIME, format="%H:%M:%S"))
NYPD_data <- NYPD_data %>%
  mutate(
    PRECINCT = as.integer(PRECINCT),
    JURISDICTION_CODE = as.integer(JURISDICTION_CODE)
  )
NYPD_data <- NYPD_data %>%
  mutate(LOCATION_DESC = as.character(LOCATION_DESC))
NYPD_data <- NYPD_data %>%
  mutate(STATISTICAL_MURDER_FLAG = as.logical(STATISTICAL_MURDER_FLAG))
NYPD_data <- NYPD_data %>%
  mutate(
    X_COORD_CD = as.numeric(X_COORD_CD),
    Y_COORD_CD = as.numeric(Y_COORD_CD),
    Latitude = as.numeric(Latitude),
    Longitude = as.numeric(Longitude)
  )
```

Identify Missing Values

This step is crucial in tidying the data - understanding which values are missing allows for further insight in how to move forward with visualization and analysis. Taking note of any specific values or patterns in the data also gives insight into which columns to keep and might be useful for analysis.

```r
#see if there are any missing values in any one of the columns (to help decide what to keep)
#will handle any missing data as well further down

missing_val <- sapply(NYPD_data, function(x) sum(x %in% c("", NA)))
missing_val_df <- data.frame(Column = names(missing_val), `Missing Values` = missing_val)
print(missing_val_df)
```

```
##                                        Column Missing.Values
## INCIDENT_KEY                     INCIDENT_KEY              0
## OCCUR_DATE                       OCCUR_DATE                0
## OCCUR_TIME                       OCCUR_TIME                0
## BORO                                   BORO                0
## LOC_OF_OCCUR_DESC          LOC_OF_OCCUR_DESC            25596
## PRECINCT                           PRECINCT                0
## JURISDICTION_CODE          JURISDICTION_CODE               2
## LOC_CLASSFCTN_DESC        LOC_CLASSFCTN_DESC            25596
## LOCATION_DESC                 LOCATION_DESC            14977
## STATISTICAL_MURDER_FLAG STATISTICAL_MURDER_FLAG            0
## PERP_AGE_GROUP               PERP_AGE_GROUP             9344
## PERP_SEX                           PERP_SEX             9310
## PERP_RACE                         PERP_RACE             9310
## VIC_AGE_GROUP                 VIC_AGE_GROUP                0
## VIC_SEX                             VIC_SEX                0
## VIC_RACE                           VIC_RACE                0
## X_COORD_CD                       X_COORD_CD                0
## Y_COORD_CD                       Y_COORD_CD                0
## Latitude                           Latitude               10
## Longitude                         Longitude               10
## Lon_Lat                             Lon_Lat               10
```

```
#how much is missing from each data type - has a little sway in choosing columns
#NOTICE LOC_OF_OCCUR_DESC BEGAN IN 2022 (LOTS OF MISSING VALUES) AND NO DESCRIPTION ON COLUMNS DESCRIPT
#LOC_DESC ALSO HAS A LOT OF BLANKS, BUT IS LISTED AS LOCATION OF SHOOTING INCIDENT
#obviously perp age, sex, race have almost equivalent blanks
#look and see occurrences in each thing to see what the columns are actually composed of
#it also helps to view csv file in Excel!
```

Select Important Columns/Variables

After exploring the data, transforming it, and identifying missing data, asking questions about the data allows for selecting the necessary columns or variables to work with. Given the initial columns, choosing variables related to location, demographic profiles of victims and perpetrators alike, as well as timeframes are all relevant topics to explore in analyzing trends in shooting incidents.

```
#eliminate columns (as seen below)
#the following lines are selecting first what columns I believe are of importance

important_vars <- c("INCIDENT_KEY", "OCCUR_DATE", "OCCUR_TIME", "BORO", "PRECINCT", "LOC_OF_OCCUR_DESC"
NYPD_data <- NYPD_data %>% select(all_of(important_vars))
```

Handling Missing Data/Tidy and Transform

Finalizing the data columns used for visualization and analysis allows for the next step of tidying and ensuring the columns are ready for manipulation. Removing any extreme or blank values is important to accurately represent the data.

```
#after checking the data again using print(NYPD_data), can identify the following missing values, etc.
#now can see the missing values in the columns we are actually keeping - can see LOC_OF_OCCUR_DESC and
#Can also see that there are empty values in LOC_OF_OCCUR_DESC, there are empty and (null) in LOCATION_

selected_columns <- c("LOC_OF_OCCUR_DESC", "LOCATION_DESC", "PERP_AGE_GROUP", "PERP_SEX", "PERP_RACE")

filtered_data <- NYPD_data %>%
  filter(across(all_of(selected_columns), ~ !(. %in% c("", NA))))
```

```
## Warning: Using ‘across()‘ in ‘filter()‘ was deprecated in dplyr 1.0.8.
## i Please use ‘if_any()‘ or ‘if_all()‘ instead.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
#can do print(filtered_data) to double-check

#the code below deals with any missing values/checks for values as empty strings (so blanks), NA, and v


NYPD_data <- NYPD_data %>%
  mutate(
    LOC_OF_OCCUR_DESC = ifelse(tolower(LOC_OF_OCCUR_DESC) %in% c("", NA, "(null)", "u", "unknown", "unkn
    LOCATION_DESC = ifelse(tolower(LOCATION_DESC) %in% c("", NA, "(null)", "u", "unknown", "unknown2"),
    PERP_AGE_GROUP = ifelse(tolower(PERP_AGE_GROUP) %in% c("", NA, "1020", "224", "940", "(null)", "u",
    PERP_SEX = ifelse(tolower(PERP_SEX) %in% c("", NA, "(null)", "u", "unknown", "unknown2"), "Unknown"
    PERP_RACE = ifelse(tolower(PERP_RACE) %in% c("", NA, "(null)", "u", "unknown", "unknown2"), "Unknow
  )
```

```r
#print(NYPD_data)

#now that all is tidied, continue with visualization and analysis
```

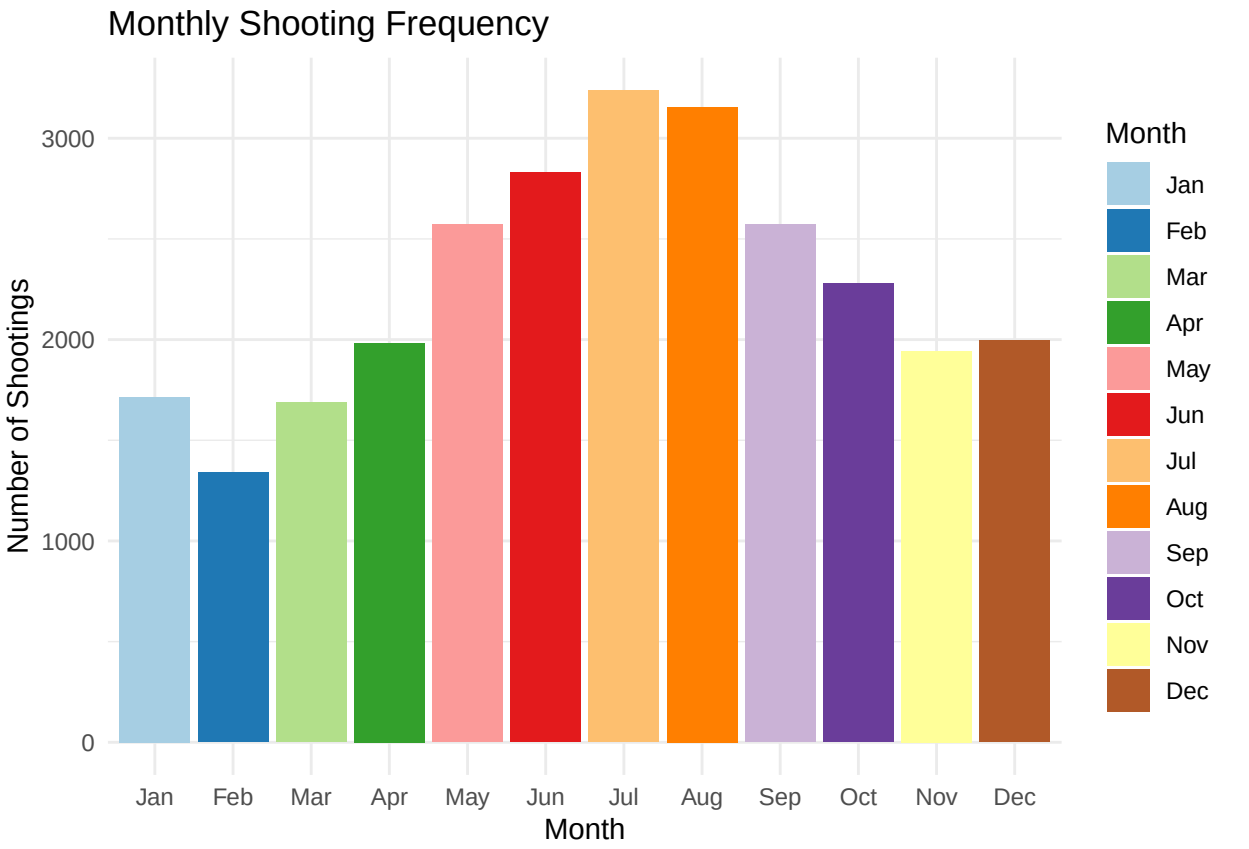
Data Visualization

After cleaning and ensuring the data columns are ready for visualization and analysis, asking the important questions that you are interested in about the dataset guides the next steps forward. In this project, I am to explore the relationship between shooting incidents and time of year, incident frequency and time of day, incidents by area (in terms of precincts and boros), murders per each boro, and the differences in demographic profiles of victims and perpetrators.

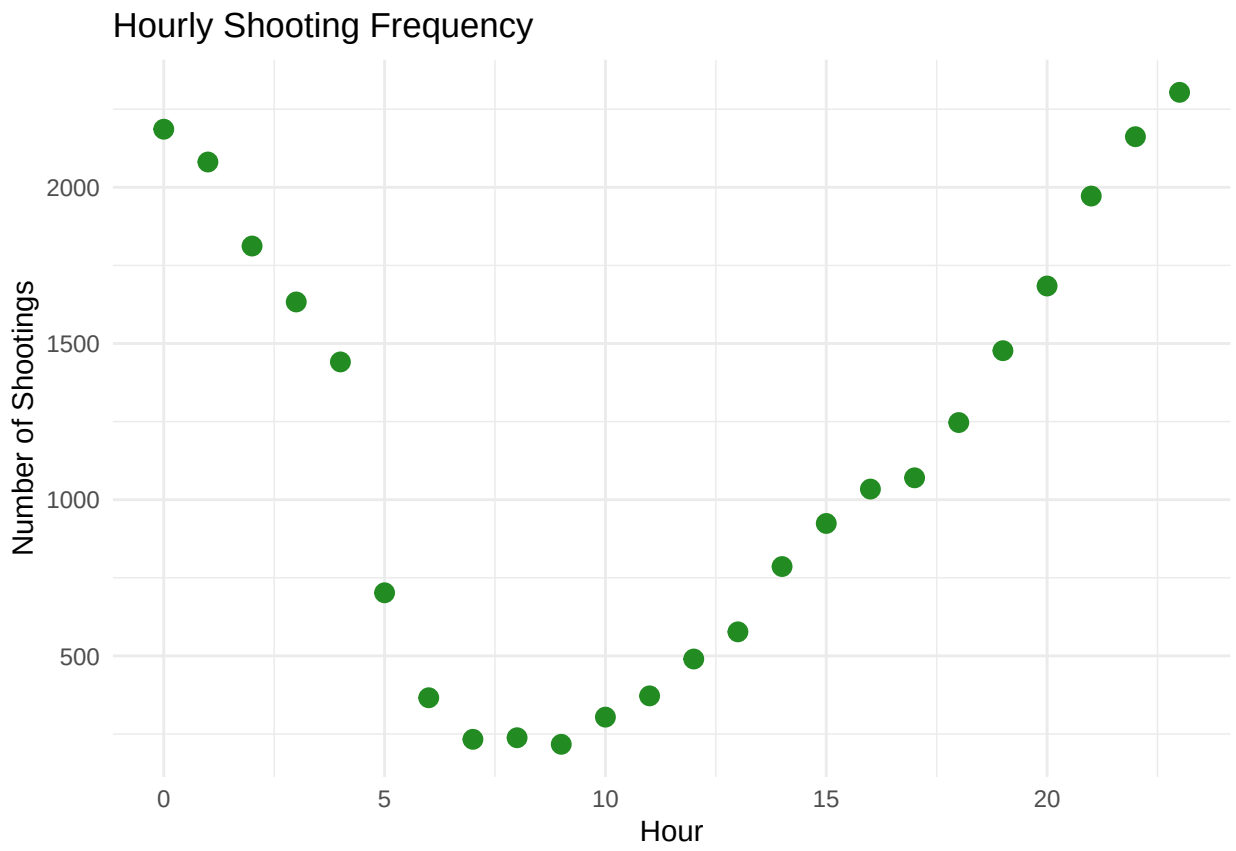
```r
#the bar chart below shows the month/time of year in which shootings occur the most frequently
month_bar_chart <- NYPD_data %>%
  group_by(Month = month(OCCUR_DATE, label = TRUE)) %>%
  summarise(ShootingCount = n()) %>%
  ggplot(aes(x = Month, y = ShootingCount, fill = Month)) +
  geom_bar(stat = "identity") +
  labs(title = "Monthly Shooting Frequency", x = "Month", y = "Number of Shootings") +
  theme_minimal() +
  scale_fill_brewer(palette = "Paired")

#the scatterplot below shows the hour of the day in which shootings occur the most frequently
hour_scatterplot <- NYPD_data %>%
     group_by(Hour = hour(OCCUR_TIME)) %>%
     summarise(ShootingCount = n()) %>%
     ggplot(aes(x = Hour, y = ShootingCount)) +
     geom_point(color = "forestgreen", size = 3) +
     labs(title = "Hourly Shooting Frequency", x = "Hour", y = "Number of Shootings") +
     theme_minimal()

print(month_bar_chart)
```




```
print(hour_scatterplot)
```

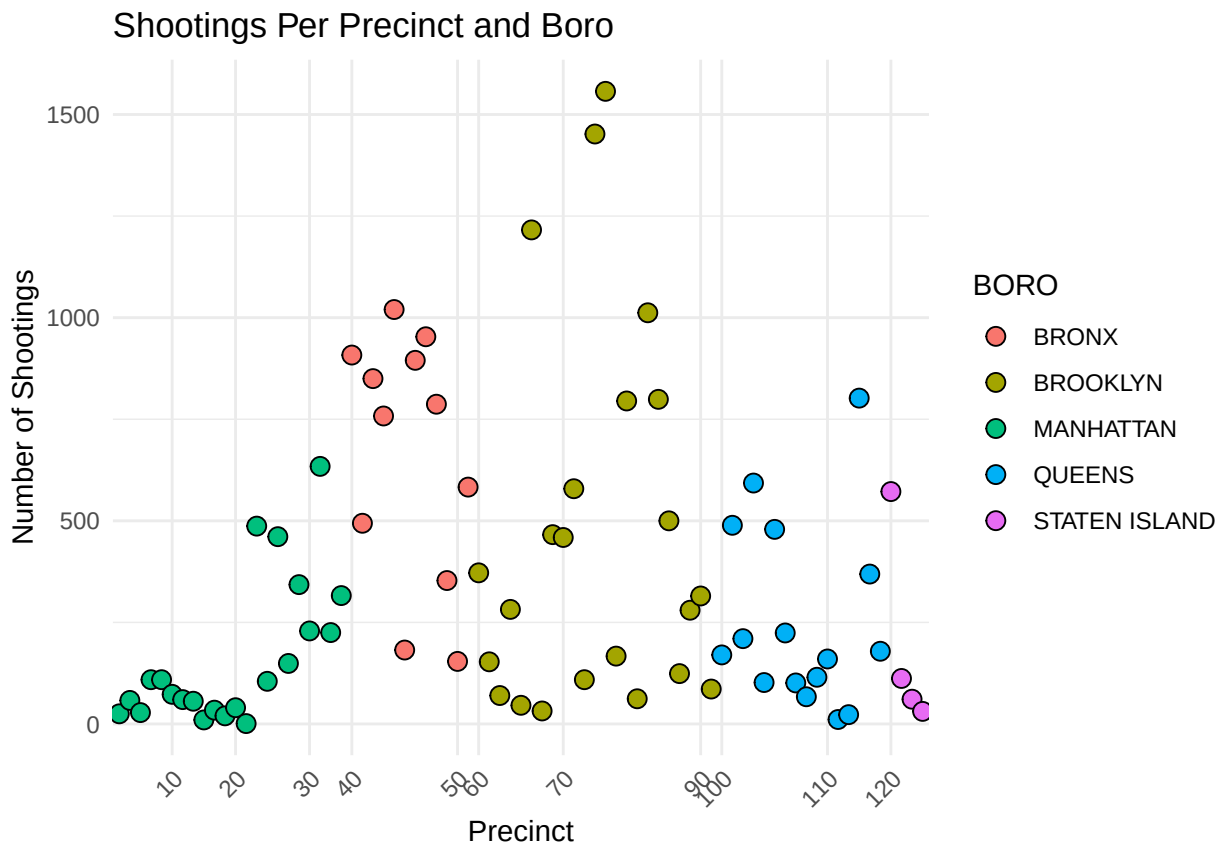



```
#This scatterplot shows the number of shootings by precinct and boro:
precinct_boro_scatterplot <- NYPD_data %>%
   group_by(BORO, PRECINCT) %>%
   summarise(Count = n()) %>%
   ggplot(aes(x = factor(PRECINCT), y = Count, fill = BORO)) +
   geom_point(position = position_dodge(width = 0.8), size = 3, shape = 21, color = "black") +
   labs(title = "Shootings Per Precinct and Boro", x = "Precinct", y = "Number of Shootings") +
   theme_minimal() +
   theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
   scale_x_discrete(breaks = seq(0, max(NYPD_data$PRECINCT), by = 10))
```

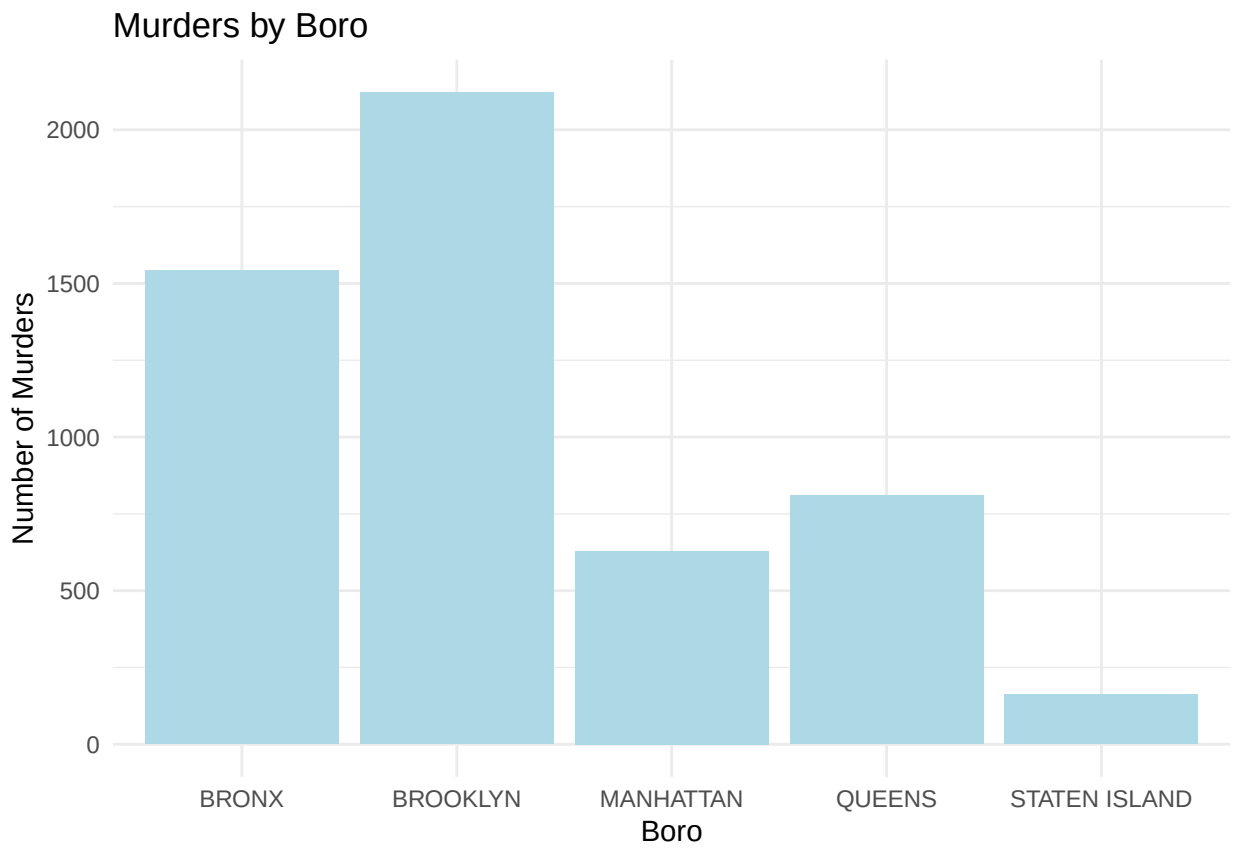
```
## `summarise()` has grouped output by 'BORO'. You can override using the
## `.groups` argument.
```

```
#This bar chart shows the number of murders per boro.
murder_boro_bar_chart <- NYPD_data %>%
     filter(STATISTICAL_MURDER_FLAG == TRUE) %>%
     group_by(BORO) %>%
     summarise(MurderCount = n()) %>%
     ggplot(aes(x = BORO, y = MurderCount)) +
     geom_bar(stat = "identity", fill = "lightblue") +
     labs(title = "Murders by Boro", x = "Boro", y = "Number of Murders") +
     theme_minimal()

print(precinct_boro_scatterplot)
```




```
print(murder_boro_bar_chart)
```



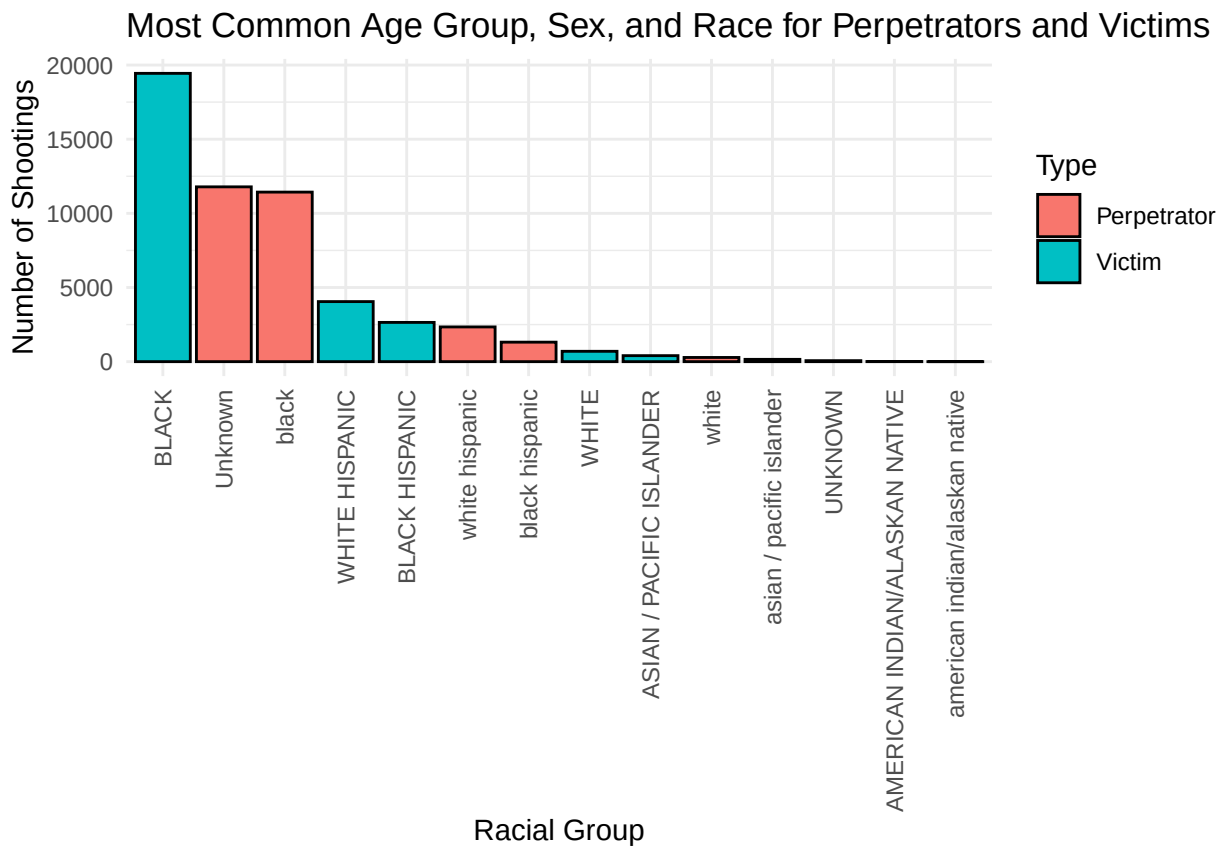

```
#This visualization shows the characteristics of both perpetrators and victims in one place for easy co

perp_and_vic <- rbind(
  NYPD_data %>%
    filter(!is.na(PERP_RACE)) %>%
    group_by(Race = PERP_RACE, Type = "Perpetrator") %>%
    summarise(Count = n()) %>%
    mutate(Type = factor(Type)),
  NYPD_data %>%
    filter(!is.na(VIC_RACE)) %>%
    group_by(Race = VIC_RACE, Type = "Victim") %>%
    summarise(Count = n()) %>%
    mutate(Type = factor(Type))
)
```

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

```
perp_and_vic_plot <- ggplot(perp_and_vic, aes(x = reorder(Race, -Count), y = Count, fill = Type)) +
  geom_bar(stat = "identity", position = "dodge", color = "black") +
  labs(title = "Most Common Age Group, Sex, and Race for Perpetrators and Victims", x = "Racial Group",
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
```

```
print(perp_and_vic_plot)
```



Data Analysis - Linear Regression

Visualzing the data allows for further insights into the dataset and leads to more questions - specifically, I chose to explore the relationship between shootings and the hour of the day by developing a linear regression model to predict shootings based on time. This analysis allows for further insight by examining the coefficients, the residual standard error indicating the deviation of observed vs predicted values for shooting counts, and the F-statistic indicating the overall significance of the model.

```
#This model shows linear regression analysis to predict shootings based on the time of day/hour
linear_reg <- lm(n ~ Hour - 1, data = NYPD_data %>%
                 group_by(Hour = hour(OCCUR_TIME)) %>%
                 summarise(n = n()))


summary(linear_reg)
```

```
##
## Call:
## lm(formula = n ~ Hour - 1, data = NYPD_data %>% group_by(Hour = hour(OCCUR_TIME)) %>%
##     summarise(n = n()))
##
## Residuals:
##    Min     1Q Median     3Q    Max
```

```
## -477.8 -325.5 -120.0  479.7 2186.0
##
## Coefficients:
##      Estimate Std. Error t value Pr(>|t|)
## Hour    77.20      13.11   5.891 5.27e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 861.8 on 23 degrees of freedom
## Multiple R-squared:  0.6014, Adjusted R-squared:  0.5841
## F-statistic:  34.7 on 1 and 23 DF,  p-value: 5.27e-06
```

```
#The model above uses the data from the data on shootings on a given hour of the day (note that the - 1
```

Bias Identification and Conclusion

Bias is an important aspect to address when analyzing and working with any type of dataset, particularly one that can be a sensitive topic for any number of people. The inherent biases I had walking into this project included the assumption that there would be a higher number of female victims of shootings, or potentially a closer margin with male victims, and that the specific boro of the Bronx would have the highest number of shootings based on my own exposure to news and other media. However, upon analysis of the data set both of these assumptions were deemed incorrect. It is also important to deal with personal biases in a broader context, and I mitigated this by cross-referencing with other news/media sources - it is interesting to know that shootings in New York have decreased significantly in recent times (approximately 25% as of June 2023). It is more important than ever to understand and analyze incoming data as trends change over time. The information uncovered in this dataset showed that the boros Brooklyn and the Bronx (and their associated precincts) were the most dangerous, particularly for males in the younger age group of 18-24. Because most shootings occurred at night and during the middle months of the year, taking precautions and moving forward with understanding these trends is highly useful. Analyzing and working with the data led me to ask further questions related to patterns in different location types, dwellings, and more specific patterns based on pepetrator and victim profiles - all of these are a solid foundation for future investigation. Overall, visualizing data in terms of perpetrator, victim, location, and time metrics and statistics allows for deeper insights that are consistent with current news and media sources.

Sources:

- "Shootings in New York Drop by a Quarter as Surge of Violence Eases" - Hurubie Meko, New York Times (https://www.nytimes.com/2023/07/06/nyregion/shootings-nyc-crime.html?auth=login-google1tap&login=google1tap)
- "NYPD Shooting Incident Data (Historic) - Data.Gov" (https://catalog.data.gov/dataset/nypd-shooting-incident-data-historic)


```
sessionInfo()
```

```
## R version 4.3.2 (2023-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
```

```
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] RColorBrewer_1.1-3 lubridate_1.9.3    forcats_1.0.0      stringr_1.5.1
##  [5] dplyr_1.1.4        purrr_1.0.2        readr_2.1.4        tidyr_1.3.0
##  [9] tibble_3.2.1       ggplot2_3.4.4      tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4      highr_0.10        crayon_1.5.2      compiler_4.3.2
##  [5] tidyselect_1.2.0  scales_1.2.1      yaml_2.3.7        fastmap_1.1.1
##  [9] R6_2.5.1          labeling_0.4.3    generics_0.1.3    knitr_1.45
## [13] munsell_0.5.0     pillar_1.9.0      tzdb_0.4.0        rlang_1.1.1
## [17] utf8_1.2.3        stringi_1.8.2     xfun_0.41         timechange_0.2.0
## [21] cli_3.6.1         withr_2.5.2       magrittr_2.0.3    digest_0.6.33
## [25] grid_4.3.2        rstudioapi_0.15.0 hms_1.1.3         lifecycle_1.0.3
## [29] vctrs_0.6.4       evaluate_0.23     glue_1.6.2        farver_2.1.1
## [33] fansi_1.0.4       colorspace_2.1-0  rmarkdown_2.25    tools_4.3.2
## [37] pkgconfig_2.0.3   htmltools_0.5.7
```