NWPD_Shooting_Incident

BQ

2024-11-24

Project Description:

In this project, we will analyze a shooting incident involving the NYPD. The data used is publicly available at "https://catalog.data.gov/dataset/nypd-shooting-incident-data-historic".

```r
# Set seed for reproducibility
set.seed(42)

# Define required packages
required_packages <- c("tidyverse", "ggplot2", "rstudioapi", "readxl", "caret")

# Check which of the required package is not installed in users' machine
need_install <- required_packages[!(required_packages) %in% installed.packages()]

# Install the required packages if any of them are not already installed
if (length(need_install) > 0 ){
  install.packages(need_install)
}


# Load packages
lapply(required_packages, require, character.only = TRUE)
```

```
## Loading required package: tidyverse

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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
## Loading required package: rstudioapi
##
## Loading required package: readxl
##
## Loading required package: caret
##
```

```
## Loading required package: lattice
##
##
## Attaching package: ’caret’
##
##
## The following object is masked from ’package:purrr’:
##
##     lift
```

```
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] TRUE
##
## [[5]]
## [1] TRUE
```

```
# Getting data
nypd_raw_data <- read_csv("https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?accessType=DOWNLO
  distinct() %>%
  drop_na()
```

```
## Rows: 28562 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (12): OCCUR_DATE, BORO, LOC_OF_OCCUR_DESC, LOC_CLASSFCTN_DESC, LOCATION...
## dbl   (7): INCIDENT_KEY, PRECINCT, JURISDICTION_CODE, X_COORD_CD, Y_COORD_CD...
## lgl   (1): STATISTICAL_MURDER_FLAG
## time  (1): OCCUR_TIME
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
df <- nypd_raw_data
```

```
# Display data structure
str(df)
```

```
## tibble [2,907 x 21] (S3: tbl_df/tbl/data.frame)
##  $ INCIDENT_KEY           : num [1:2907] 2.45e+08 2.48e+08 2.55e+08 2.50e+08 2.43e+08 ...
##  $ OCCUR_DATE             : chr [1:2907] "05/05/2022" "07/04/2022" "11/30/2022" "08/15/2022" ...
##  $ OCCUR_TIME             : ’hms’ num [1:2907] 00:10:00 22:20:00 21:15:00 18:21:00 ...
##   ..- attr(*, "units")= chr "secs"
##  $ BORO                   : chr [1:2907] "MANHATTAN" "BRONX" "BRONX" "QUEENS" ...
##  $ LOC_OF_OCCUR_DESC      : chr [1:2907] "INSIDE" "OUTSIDE" "OUTSIDE" "OUTSIDE" ...
##  $ PRECINCT               : num [1:2907] 14 48 46 101 49 75 49 121 9 69 ...
##  $ JURISDICTION_CODE      : num [1:2907] 0 0 0 2 0 0 0 0 2 0 ...
##  $ LOC_CLASSFCTN_DESC     : chr [1:2907] "COMMERCIAL" "STREET" "STREET" "HOUSING" ...
##  $ LOCATION_DESC          : chr [1:2907] "VIDEO STORE" "(null)" "(null)" "MULTI DWELL - PUBLIC HOUS"
##  $ STATISTICAL_MURDER_FLAG: logi [1:2907] TRUE TRUE TRUE TRUE FALSE TRUE ...
##  $ PERP_AGE_GROUP         : chr [1:2907] "25-44" "(null)" "18-24" "(null)" ...
##  $ PERP_SEX               : chr [1:2907] "M" "(null)" "M" "(null)" ...
##  $ PERP_RACE              : chr [1:2907] "BLACK" "(null)" "BLACK" "(null)" ...
##  $ VIC_AGE_GROUP          : chr [1:2907] "25-44" "18-24" "<18" "18-24" ...
##  $ VIC_SEX                : chr [1:2907] "M" "M" "M" "M" ...
##  $ VIC_RACE               : chr [1:2907] "BLACK" "BLACK" "BLACK" "BLACK" ...
##  $ X_COORD_CD             : num [1:2907] 986050 1016802 1011263 1053494 1021686 ...
##  $ Y_COORD_CD             : num [1:2907] 214231 250581 251671 161531 251947 ...
##  $ Latitude               : num [1:2907] 40.8 40.9 40.9 40.6 40.9 ...
##  $ Longitude              : num [1:2907] -74 -73.9 -73.9 -73.8 -73.9 ...
##  $ Lon_Lat                : chr [1:2907] "POINT (-73.9935 40.754692)" "POINT (-73.88233 40.854402)"
```

```
summary(df)
```

Summary of the data frame

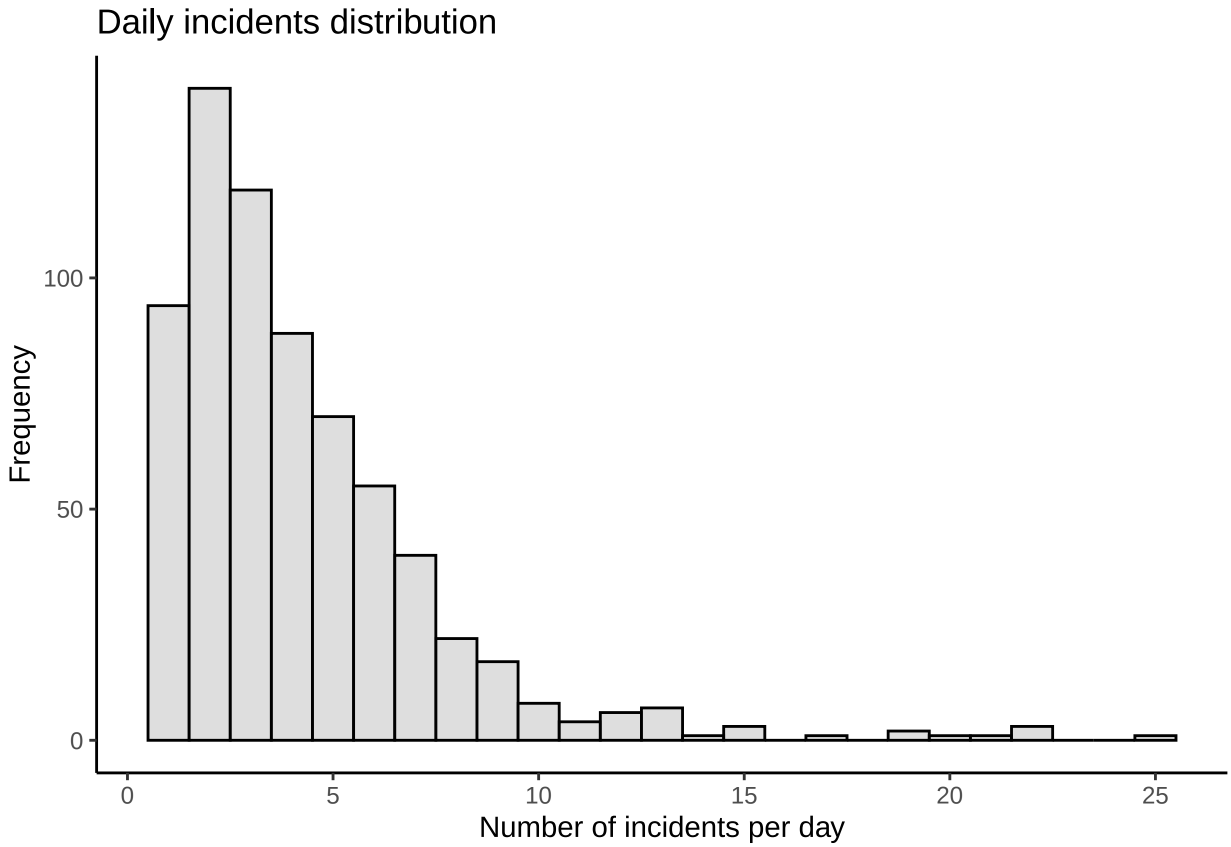
```
##   INCIDENT_KEY        OCCUR_DATE         OCCUR_TIME           BORO
##  Min.   :238531159   Length:2907        Length:2907       Length:2907
##  1st Qu.:246192328   Class :character   Class1:hms        Class :character
##  Median :252647955   Mode  :character   Class2:difftime   Mode  :character
##  Mean   :256854604                      Mode  :numeric
##  3rd Qu.:268973603
##  Max.   :279758069
##  LOC_OF_OCCUR_DESC     PRECINCT      JURISDICTION_CODE LOC_CLASSFCTN_DESC
##  Length:2907        Min.   :  1.00   Min.   :0.0000    Length:2907
##  Class :character   1st Qu.: 43.00   1st Qu.:0.0000    Class :character
##  Mode  :character   Median : 60.00   Median :0.0000    Mode  :character
##                     Mean   : 62.22   Mean   :0.2425
##                     3rd Qu.: 79.00   3rd Qu.:0.0000
##                     Max.   :123.00   Max.   :2.0000
##  LOCATION_DESC      STATISTICAL_MURDER_FLAG PERP_AGE_GROUP
##  Length:2907        Mode :logical           Length:2907
##  Class :character   FALSE:2313              Class :character
##  Mode  :character   TRUE :594               Mode  :character
##
##
##
##    PERP_SEX           PERP_RACE         VIC_AGE_GROUP        VIC_SEX
##  Length:2907        Length:2907        Length:2907        Length:2907
##  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:2907        Min.   : 929510   Min.   :127539   Min.   :40.52  
##  Class :character   1st Qu.:1000459   1st Qu.:184337   1st Qu.:40.67  
##  Mode  :character   Median :1008366   Median :212367   Median :40.75  
##                     Mean   :1009286   Mean   :212612   Mean   :40.75  
##                     3rd Qu.:1016743   3rd Qu.:242614   3rd Qu.:40.83  
##                     Max.   :1059828   Max.   :269204   Max.   :40.91  
##    Longitude        Lon_Lat         
##  Min.   :-74.20   Length:2907       
##  1st Qu.:-73.94   Class :character  
##  Median :-73.91   Mode  :character  
##  Mean   :-73.91                     
##  3rd Qu.:-73.88                     
##  Max.   :-73.73
```

```
head(df)
```

```
## # A tibble: 6 x 21
##   INCIDENT_KEY OCCUR_DATE OCCUR_TIME BORO      LOC_OF_OCCUR_DESC PRECINCT
##          <dbl> <chr>      <time>     <chr>     <chr>                <dbl>
## 1    244608249 05/05/2022 00:10      MANHATTAN INSIDE                  14
## 2    247542571 07/04/2022 22:20      BRONX     OUTSIDE                 48
## 3    254911480 11/30/2022 21:15      BRONX     OUTSIDE                 46
## 4    249623757 08/15/2022 18:21      QUEENS    OUTSIDE                101
## 5    243433246 04/10/2022 17:00      BRONX     OUTSIDE                 49
## 6    253757468 11/07/2022 11:35      BROOKLYN  OUTSIDE                 75
## # i 15 more variables: JURISDICTION_CODE <dbl>, LOC_CLASSFCTN_DESC <chr>,
## #   LOCATION_DESC <chr>, STATISTICAL_MURDER_FLAG <lgl>, PERP_AGE_GROUP <chr>,
## #   PERP_SEX <chr>, PERP_RACE <chr>, VIC_AGE_GROUP <chr>, VIC_SEX <chr>,
## #   VIC_RACE <chr>, X_COORD_CD <dbl>, Y_COORD_CD <dbl>, Latitude <dbl>,
## #   Longitude <dbl>, Lon_Lat <chr>
```

Data exploration, data cleaning and trasnformation

```
# Getting count of daily incidents
daily_count <- df %>%
  group_by(OCCUR_DATE) %>%
  summarise(INCIDENT_COUNT = n(), .groups = "drop")

ggplot(daily_count, aes(x = INCIDENT_COUNT)) +
  geom_histogram(binwidth = 1, fill = "grey", color = "black", alpha = 0.5) +
  labs(title = "Daily incidents distribution",
       x = "Number of incidents per day",
       y = "Frequency") +
  theme_classic()
```



```r
df <- df %>%
  mutate(
    OCCUR_DATE = mdy(OCCUR_DATE),
    Year = year(OCCUR_DATE),
    Month = factor(month(OCCUR_DATE, label = TRUE, abbr = TRUE), levels = month.abb),
    DayOfWeek = factor(wday(OCCUR_DATE, label = TRUE, abbr = TRUE), levels = c("Sun", "Mon", "Tue", "We
    TimeOfDay = case_when(
      hour(OCCUR_TIME) >= 6 & hour(OCCUR_TIME) < 12 ~ "Morning",
      hour(OCCUR_TIME) >= 12 & hour(OCCUR_TIME) < 18 ~ "Afternoon",
      hour(OCCUR_TIME) >= 18 & hour(OCCUR_TIME) < 24 ~ "Evening",
      TRUE ~ "Night"
    )
  )
```

```r
# Checking the new data structure
str(df)
```

```
## tibble [2,907 x 25] (S3: tbl_df/tbl/data.frame)
##  $ INCIDENT_KEY           : num [1:2907] 2.45e+08 2.48e+08 2.55e+08 2.50e+08 2.43e+08 ...
##  $ OCCUR_DATE             : Date[1:2907], format: "2022-05-05" "2022-07-04" ...
##  $ OCCUR_TIME             : 'hms' num [1:2907] 00:10:00 22:20:00 21:15:00 18:21:00 ...
##   ..- attr(*, "units")= chr "secs"
##  $ BORO                   : chr [1:2907] "MANHATTAN" "BRONX" "BRONX" "QUEENS" ...
##  $ LOC_OF_OCCUR_DESC      : chr [1:2907] "INSIDE" "OUTSIDE" "OUTSIDE" "OUTSIDE" ...
##  $ PRECINCT               : num [1:2907] 14 48 46 101 49 75 49 121 9 69 ...
##  $ JURISDICTION_CODE      : num [1:2907] 0 0 0 2 0 0 0 0 2 0 ...
##  $ LOC_CLASSFCTN_DESC     : chr [1:2907] "COMMERCIAL" "STREET" "STREET" "HOUSING" ...
##  $ LOCATION_DESC          : chr [1:2907] "VIDEO STORE" "(null)" "(null)" "MULTI DWELL - PUBLIC HOUS"
##  $ STATISTICAL_MURDER_FLAG: logi [1:2907] TRUE TRUE TRUE TRUE FALSE TRUE ...
##  $ PERP_AGE_GROUP         : chr [1:2907] "25-44" "(null)" "18-24" "(null)" ...
##  $ PERP_SEX               : chr [1:2907] "M" "(null)" "M" "(null)" ...
##  $ PERP_RACE              : chr [1:2907] "BLACK" "(null)" "BLACK" "(null)" ...
##  $ VIC_AGE_GROUP          : chr [1:2907] "25-44" "18-24" "<18" "18-24" ...
##  $ VIC_SEX                : chr [1:2907] "M" "M" "M" "M" ...
##  $ VIC_RACE               : chr [1:2907] "BLACK" "BLACK" "BLACK" "BLACK" ...
##  $ X_COORD_CD             : num [1:2907] 986050 1016802 1011263 1053494 1021686 ...
##  $ Y_COORD_CD             : num [1:2907] 214231 250581 251671 161531 251947 ...
##  $ Latitude               : num [1:2907] 40.8 40.9 40.9 40.6 40.9 ...
##  $ Longitude              : num [1:2907] -74 -73.9 -73.9 -73.8 -73.9 ...
##  $ Lon_Lat                : chr [1:2907] "POINT (-73.9935 40.754692)" "POINT (-73.88233 40.854402)"
##  $ Year                   : num [1:2907] 2022 2022 2022 2022 2022 ...
##  $ Month                  : Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 5 7 11 8 4 11 12 6 10 2 ..
##  $ DayOfWeek              : Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 5 2 4 2 1 2 7 1 5 3 ...
##  $ TimeOfDay              : chr [1:2907] "Night" "Evening" "Evening" "Evening" ...
```

```
summary(df)
```

```
##   INCIDENT_KEY        OCCUR_DATE          OCCUR_TIME           BORO
##  Min.   :238531159   Min.   :2022-01-01   Length:2907       Length:2907
##  1st Qu.:246192328   1st Qu.:2022-06-06   Class1:hms        Class :character
##  Median :252647955   Median :2022-10-15   Class2:difftime   Mode  :character
##  Mean   :256854604   Mean   :2022-11-24   Mode  :numeric
##  3rd Qu.:268973603   3rd Qu.:2023-05-28
##  Max.   :279758069   Max.   :2023-12-29
##
##  LOC_OF_OCCUR_DESC     PRECINCT      JURISDICTION_CODE LOC_CLASSFCTN_DESC
##  Length:2907        Min.   :  1.00   Min.   :0.0000    Length:2907
##  Class :character   1st Qu.: 43.00   1st Qu.:0.0000    Class :character
##  Mode  :character   Median : 60.00   Median :0.0000    Mode  :character
##                     Mean   : 62.22   Mean   :0.2425
##                     3rd Qu.: 79.00   3rd Qu.:0.0000
##                     Max.   :123.00   Max.   :2.0000
##
##  LOCATION_DESC      STATISTICAL_MURDER_FLAG PERP_AGE_GROUP
##  Length:2907        Mode :logical           Length:2907
##  Class :character   FALSE:2313              Class :character
##  Mode  :character   TRUE :594               Mode  :character
##
##
##
##
##    PERP_SEX           PERP_RACE          VIC_AGE_GROUP        VIC_SEX
##  Length:2907        Length:2907        Length:2907        Length:2907
##  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:2907        Min.   : 929510   Min.   :127539   Min.   :40.52  
##  Class :character   1st Qu.:1000459   1st Qu.:184337   1st Qu.:40.67  
##  Mode  :character   Median :1008366   Median :212367   Median :40.75  
##                     Mean   :1009286   Mean   :212612   Mean   :40.75  
##                     3rd Qu.:1016743   3rd Qu.:242614   3rd Qu.:40.83  
##                     Max.   :1059828   Max.   :269204   Max.   :40.91  
##                                                                       
##    Longitude        Lon_Lat               Year          Month        DayOfWeek
##  Min.   :-74.20   Length:2907        Min.   :2022   Jul    : 375   Sun:502  
##  1st Qu.:-73.94   Class :character   1st Qu.:2022   Jun    : 290   Mon:451  
##  Median :-73.91   Mode  :character   Median :2022   May    : 277   Tue:370  
##  Mean   :-73.91                      Mean   :2022   Mar    : 262   Wed:323  
##  3rd Qu.:-73.88                      3rd Qu.:2023   Aug    : 259   Thu:358  
##  Max.   :-73.73                      Max.   :2023   Sep    : 254   Fri:372  
##                                                     (Other):1190   Sat:531  
##   TimeOfDay        
##  Length:2907       
##  Class :character  
##  Mode  :character  
##                    
##                    
##                    
## 
```

Grouping and Summarization

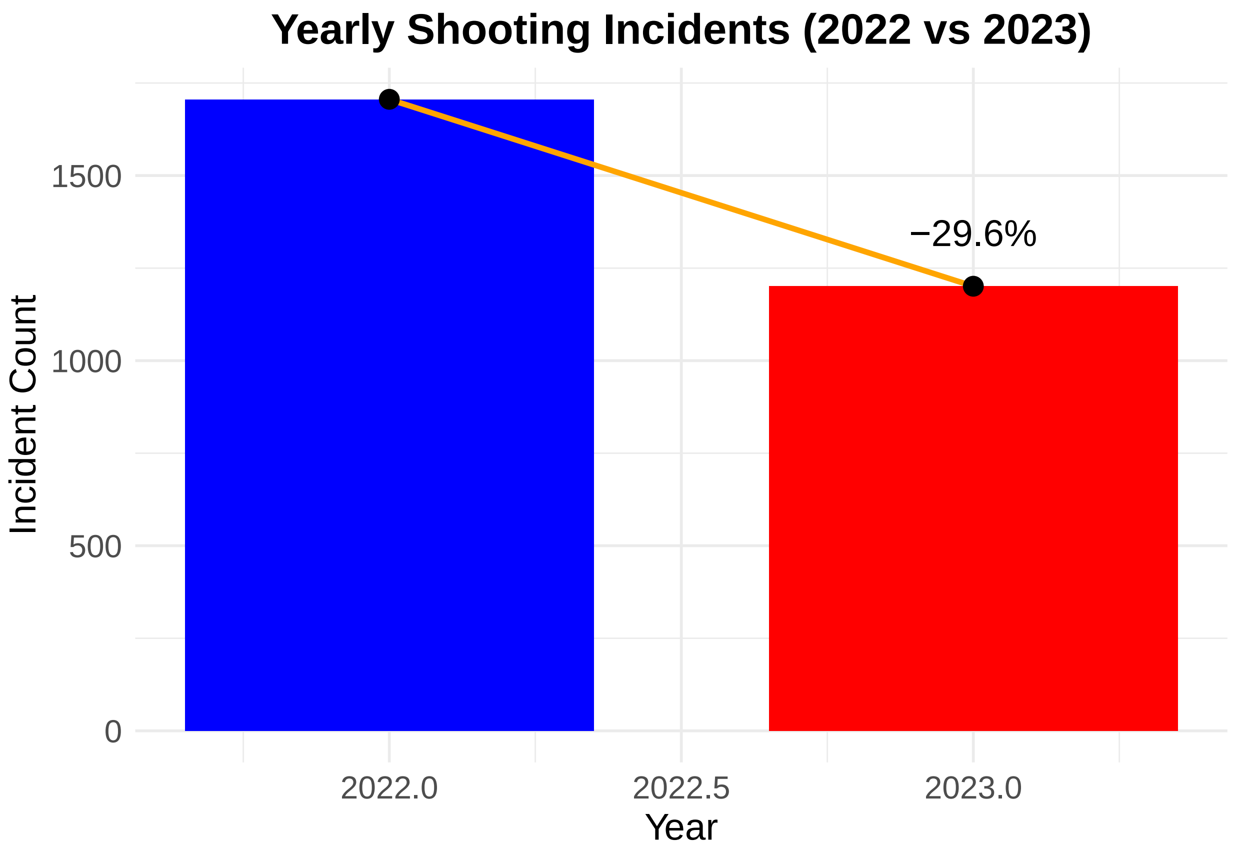
Total Incident for each year We group the data by ‘Year’ to count total number of of incidents for each year. This summarized data will be used later to create a bar chart comparing the yearly totals.

```r
# Group by Year and count total incidents
yearly_incidents <- df %>%
  group_by(Year) %>%
  summarise(Incident_Count = n(), .groups = "drop")

# View the results
print(yearly_incidents)
```

```
## # A tibble: 2 x 2
##    Year Incident_Count
##   <dbl>          <int>
## 1  2022           1706
## 2  2023           1201
```

```r
# Add a percentage change column for annotations
yearly_incidents <- yearly_incidents %>%
  mutate(Percent_Change = c(NA, (Incident_Count[2] - Incident_Count[1]) / Incident_Count[1] * 100))

# Create the combined bar and line chart
ggplot(yearly_incidents, aes(x = Year)) +
  # Bar chart for incident counts
  geom_bar(aes(y = Incident_Count, fill = as.factor(Year)), stat = "identity", width = 0.7, show.legend
  # Line plot for percentage change
  geom_line(aes(y = Incident_Count, group = 1), color = "orange", size = 1) +
  geom_point(aes(y = Incident_Count), color = "black", size = 3) +
  # Add percentage labels on the line
  geom_text(aes(y = Incident_Count, label = ifelse(is.na(Percent_Change), "", paste0(round(Percent_Chang
            vjust = -1.5, size = 5, color = "black", na.rm = TRUE) +
  # labels and theme
  labs(
    title = "Yearly Shooting Incidents (2022 vs 2023)",
    x = "Year",
    y = "Incident Count",
    fill = "Year"
  ) +
  scale_fill_manual(values = c("2022" = "blue", "2023" = "red")) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12)
  )
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



Month to Month comparison We will group the data by the Year and Month and count the incident for each combination, than will plot a monthly trends for 2022 and 2023 as a two separate lines for comparison. {r monthly-counts} # Group by Year and Month, and count incidents monthly_incidents <- df %>% group_by(Year, Month) %>% summarise(Incident_Count = n(), .groups = "drop") %>%

View results

print(monthly_incidents)

{r monthly-counts-chart} # Create the line chart ggplot(monthly_incidents, aes(x = Month, y = Incident_Count, colour = as.factor(Year), group = Year)) + geom_line(size = 1) + geom_point(size = 2) + labs(title = "Monthly Shooting incidents: 2022 vs 2023", x = "Months", y = "Incident Count", color = "Years") + scale_color_manual(values = c("2022" = "blue", "2023" = "red")) + theme_minimal() + theme(plot.title = element_text(size = 16, face = "bold", hjust = 0.5), axis.title.x = element_text(size = 14), axis.title.y = element_text(size = 14), axis.text.x = element_text(size = 12, angle = 45, hjust = 1), axis.text.y = element_text(size = 12), legend.title = element_text(size = 12), legend.text = element_text(size = 10))

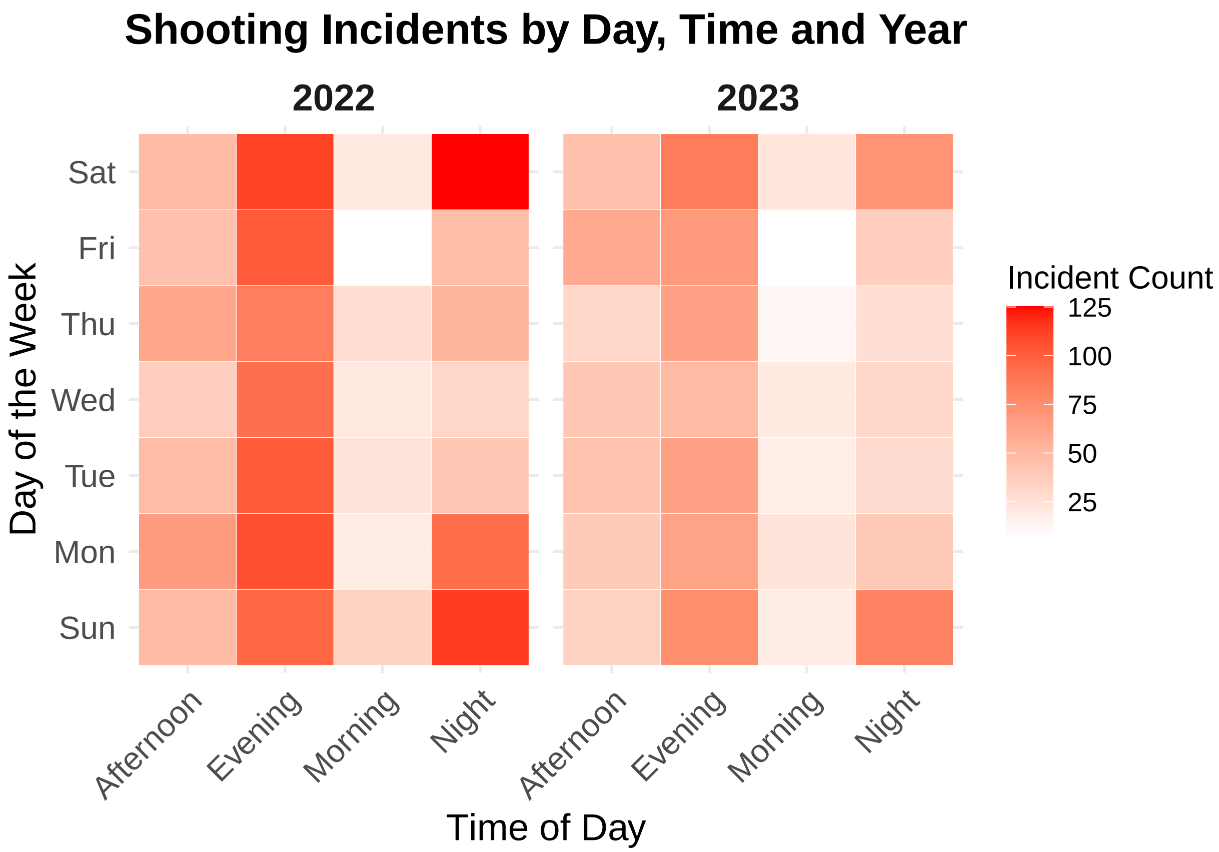
The Blue line (2022) consistently shows higher incident counts thatn the red line (2023) accoss most months The peak in 2022 occurs in July, with more than 200 incidents 2023, while having fewer incidents overall, also shows a slight peak in the summer months (July-August) 2023 shows consistently lower counts compared to 2022, with 29.6% decline overall Interestingly, the counts for November and December are very close in both years, indicating a leveling off of the decline towards the end of the year

```r
# Group by DayOfWeek and count incidents
day_of_week_analysis <- df %>%
  group_by(Year, DayOfWeek, TimeOfDay) %>%
  summarise(Incident_Count = n(), .groups = "drop")

# View the data
print(day_of_week_analysis)
```

Incident by day of the week, yearly comperison

```
## # A tibble: 56 x 4
##     Year DayOfWeek TimeOfDay Incident_Count
##    <dbl> <ord>     <chr>              <int>
##  1  2022 Sun       Afternoon             49
##  2  2022 Sun       Evening               96
##  3  2022 Sun       Morning               34
##  4  2022 Sun       Night                114
##  5  2022 Mon       Afternoon             67
##  6  2022 Mon       Evening              106
##  7  2022 Mon       Morning               19
##  8  2022 Mon       Night                 93
##  9  2022 Tue       Afternoon             48
## 10  2022 Tue       Evening              102
## # i 46 more rows
```

```r
# Create the bar chart
ggplot(day_of_week_analysis, aes(x = TimeOfDay, y = DayOfWeek, fill = Incident_Count)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "red") +
  facet_wrap(~ Year) +
  labs(
    title = "Shooting Incidents by Day, Time and Year",
    x = "Time of Day",
    y = "Day of the Week",
    fill = "Incident Count"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text.x = element_text(size = 12, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 12),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10),
    strip.text = element_text(size = 14, face = "bold")
  )
```



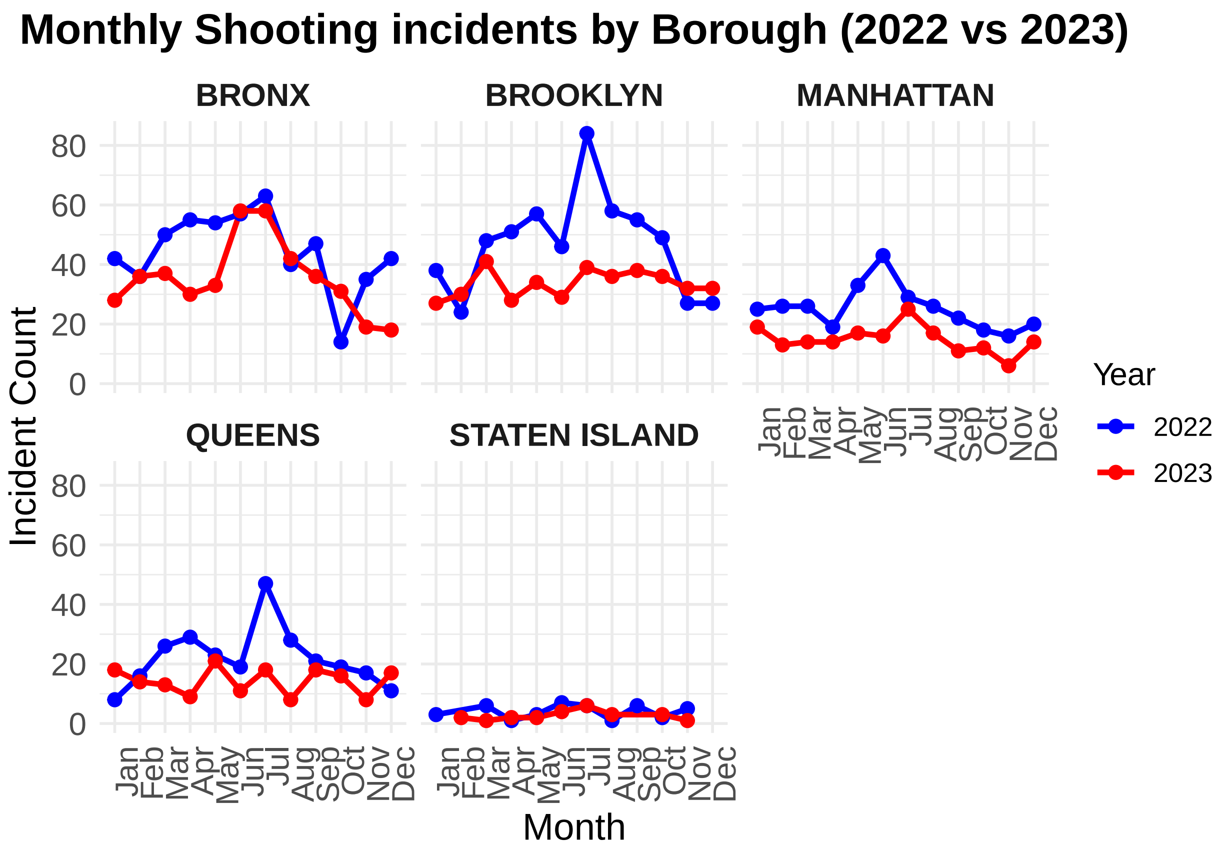
Incidents are consistently high during the night across most days of the week Saturdays and Sundays show particularly high incidents during the night in both years, with strong activity in the evening as well, which could be due to social gatherings The heatmap for 2023 is generally lighter color than 2022, indicating fewer incidents accross all times and days, which aligns with previous analyses

```r
# Group data by Month, Year and BORO
boro_monthly <- df %>%
  group_by(Year, Month, BORO) %>%
  summarise(Incident_Count = n(), .groups = "drop")

# View Monthly data
print(boro_monthly)
```

Count of Incident per Borough, Monthly and Yearly comparison

```
## # A tibble: 115 x 4
##     Year Month BORO          Incident_Count
##    <dbl> <ord> <chr>                  <int>
##  1  2022 Jan   BRONX                     42
##  2  2022 Jan   BROOKLYN                  38
##  3  2022 Jan   MANHATTAN                 25
##  4  2022 Jan   QUEENS                     8
##  5  2022 Jan   STATEN ISLAND              3
##  6  2022 Feb   BRONX                     36
##  7  2022 Feb   BROOKLYN                   24
##  8  2022 Feb   MANHATTAN                  26
##  9  2022 Feb   QUEENS                     16
## 10  2022 Mar   BRONX                      50
## # i 105 more rows
```

```
# Create faceted line plots for monthly trends
ggplot(boro_monthly, aes(x = Month, y = Incident_Count, color = as.factor(Year), group = Year)) +
  geom_line(size = 1) +
  geom_point(size = 2) +
  facet_wrap(~ BORO) +
  labs(
    title = "Monthly Shooting incidents by Borough (2022 vs 2023)",
    x = "Month",
    y = "Incident Count",
    color = "Year"
  ) +
  scale_color_manual(values = c("2022" = "blue", "2023" = "red")) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text.x = element_text(size = 12, angle = 90, hjust = 1),
    axis.text.y = element_text(size = 12),
    strip.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10)
  )
```



Across all boroughs, incidents tent to peak in the summer months (June - August) and this seasonal pattern is consistent in both years Incidents in 2023 (red line) are consistently lower than in 2022 (blue line) across most months and boroughs, aligning with the overall decline BROOKLYN and BRONX have the highest number of incidents compared to other boroughs, with noticeable peaks in July for both years STATEN ISLAND consistently has the lowest number of incidents with almost no seasonal variations. also the difference between 2022 and 2023 is minimal, indicating stability in this borough The decline in 2023 is evident in all boroughs, but the Bronx and Brooklyn contribute the most to the overall decline

Machine Learning Model

Linear Regression Model We will build a Linear Regression model to predict the number of shooting incidents (Incident_Count) for each borough (BORO) in a a specific month (Month) and year (Year).

```
# Aggregate data by BORO, Month, Year
data_model <- df %>%
  group_by(BORO, Year, Month) %>%
  summarise(Incident_Count = n(), .groups = "drop") %>%
  mutate(BORO = as.factor(BORO))

print(head(data_model))
```

```
## # A tibble: 6 x 4
##   BORO   Year Month Incident_Count
##   <fct> <dbl> <ord>          <int>
## 1 BRONX  2022 Jan               42
## 2 BRONX  2022 Feb               36
## 3 BRONX  2022 Mar              50
## 4 BRONX  2022 Apr              55
## 5 BRONX  2022 May              54
## 6 BRONX  2022 Jun              57
```

Train-Test Split We will split the data into training(80%) and testing(20%) subsets to evaluate the model's performance

```
#Split data into training and testing sets
set.seed(42) # for reproducibility
train_index <- createDataPartition(data_model$Incident_Count, p=0.8, list = FALSE)
train_data <- data_model[train_index,] # the 80% portion
test_data <- data_model[-train_index,] # the remaining 20% portion

# View the size of the train and test datasets
cat("Training data:", nrow(train_data), "\nTesting data:", nrow(test_data))
```

```
## Training data: 93
## Testing data: 22
```

Fit a Linear Regression Model We will train a simple Linear Regression model using lm() function with BORO, Month, and Year as predictor

```
# Fit the linear regression model
lm_model <- lm(Incident_Count ~ BORO + Month + Year, data = train_data)

# View the model summary
summary(lm_model)
```

```
##
## Call:
## lm(formula = Incident_Count ~ BORO + Month + Year, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -25.1596  -4.2528   0.1821   4.8947  26.0395
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       17420.3180  3569.4956   4.880 5.70e-06 ***
## BOROBROOKLYN          2.6973     2.7087   0.996   0.3225
## BOROMANHATTAN       -18.3250     2.7139  -6.752 2.56e-09 ***
## BOROQUEENS          -20.3417     2.7093  -7.508 9.57e-11 ***
## BOROSTATEN ISLAND   -36.9258     2.9752 -12.411  < 2e-16 ***
## Month.L              -3.6049     3.0803  -1.170   0.2455
## Month.Q             -15.7600     3.1496  -5.004 3.53e-06 ***
## Month.C               0.4854     3.1162   0.156   0.8766
## Month^4               6.5794     3.0328   2.169   0.0332 *
## Month^5               2.7303     2.9869   0.914   0.3636
## Month^6              -0.5697     2.9592  -0.193   0.8478
## Month^7              -2.0488     2.9528  -0.694   0.4899
## Month^8               4.6782     2.9766   1.572   0.1202
## Month^9              -0.3467     2.9751  -0.117   0.9075
## Month^10             -1.3182     3.0224  -0.436   0.6640
## Month^11             -5.3750     3.0723  -1.750   0.0842 .
## Year                 -8.5942     1.7649  -4.870 5.94e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.304 on 76 degrees of freedom
## Multiple R-squared:  0.8053, Adjusted R-squared:  0.7643
## F-statistic: 19.65 on 16 and 76 DF,  p-value: < 2.2e-16
```

Model Prediction We will use the trained modelt to predict Incident_Count on the test data

```
# Predict on the test data
predictions <- predict(lm_model, newdata = test_data)

# Combine the predections with actual values
result <- data.frame(
  Actual = test_data$Incident_Count,
  Predicted = predictions
)

print(head(result))
```

```
##   Actual Predicted
## 1     42  38.40421
## 2     55  42.84694
## 3     63  55.26316
## 4     37  36.51108
## 5     58  46.66897
## 6     27  32.50733
```

Actual values are a bit higher than Predicted values in the 6 rows of data, we will quantify how close these predictions are to the actual values using MAE and MSE

```
# Calculate MAE and MSE
mae <- mean(abs(result$Actual - result$Predicted))
mse <- mean((result$Actual - result$Predicted)^2)

# view the evaluation
cat("MAE: ", mae, "\n")
```

```
## MAE:  5.179312
```

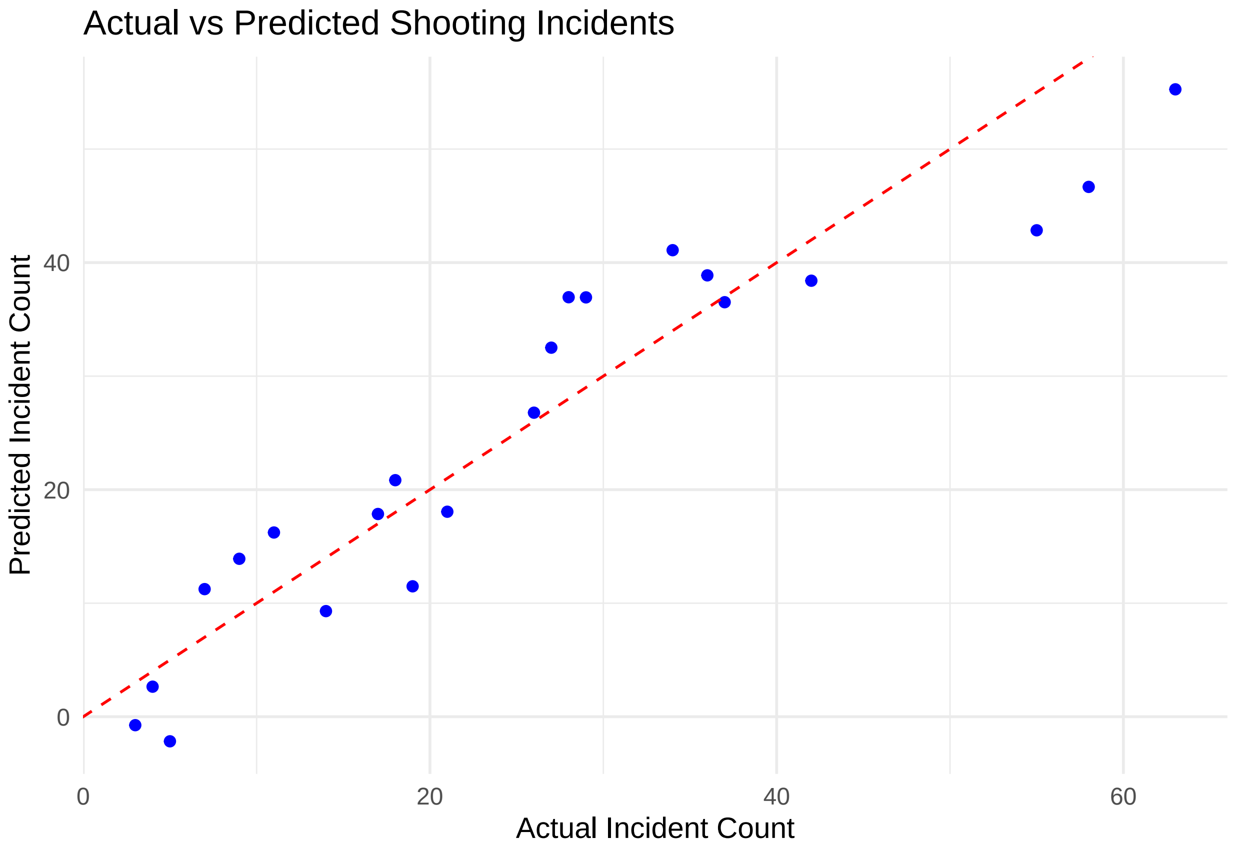
```
cat("MSE: ", mse, "\n")
```

```
## MSE:  37.01763
```

MAE value suggests the model performs reasonably well but has room for improvement MSE value highlights that there may still be occasional large errors in prediction

Result Visualization We will create a scatter plot to visualize the relationship between actual and predicted values

```
# Scatter plot of actual vs. predicted values
ggplot(result, aes(x = Actual, y = Predicted)) +
  geom_point(color = "blue") +
  geom_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +
  labs(
    title = "Actual vs Predicted Shooting Incidents",
    x = "Actual Incident Count",
    y = "Predicted Incident Count"
  )+
  theme_minimal()
```


Actual vs Predicted Shooting Incidents
Predicted Incident Count
Actual Incident Count