

# NY Shooting Incident Data Assignment

CS

## Assignment

Import, tidy and analyze the NYPD Shooting Incident dataset obtained. Be sure your project is reproducible and contains some visualization and analysis. You may use the data to do any analysis that is of interest to you. You should include at least two visualizations and one model. Be sure to identify any bias possible in the data and in your analysis.

```
library(tidyverse)
library(lubridate)
```

```
-- 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
```

```
# import the source data and put it in a df
```

```
source_url <- paste0(
  "https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?",
  "accessType=DOWNLOAD"
)
```

```
incident_df <- read.csv(source_url)
```

## Explore

```
# see what columns we have and what data types
glimpse(incident_df)
```

```
Rows: 28,562
Columns: 21
$ INCIDENT_KEY      <int> 244608249, 247542571, 84967535, 202853370, 270~
$ OCCUR_DATE        <chr> "05/05/2022", "07/04/2022", "05/27/2012", "09/~
$ OCCUR_TIME        <chr> "00:10:00", "22:20:00", "19:35:00", "21:00:00"~
$ BORO              <chr> "MANHATTAN", "BRONX", "QUEENS", "BRONX", "BROO~
$ LOC_OF_OCCUR_DESC  <chr> "INSIDE", "OUTSIDE", "", "", "", "", "", "", ""~
$ PRECINCT          <int> 14, 48, 103, 42, 83, 23, 113, 77, 48, 49, 73, ~
$ JURISDICTION_CODE <int> 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0~
$ LOC_CLASSFCTN_DESC <chr> "COMMERCIAL", "STREET", "", "", "", "", "", ""~
$ LOCATION_DESC      <chr> "VIDEO STORE", "(null)", "", "", "", "MULTI DW~
$ STATISTICAL_MURDER_FLAG <chr> "true", "true", "false", "false", "false", "fa~
$ PERP_AGE_GROUP     <chr> "25-44", "(null)", "", "25-44", "25-44", "", "~
$ PERP_SEX           <chr> "M", "(null)", "", "M", "M", "", "", "", "", "~
$ PERP_RACE           <chr> "BLACK", "(null)", "", "UNKNOWN", "BLACK", "",~
$ VIC_AGE_GROUP      <chr> "25-44", "18-24", "18-24", "25-44", "25-44", "~
$ VIC_SEX             <chr> "M", "M", "M", "M", "M", "M", "M", "M", "M", "~
$ VIC_RACE             <chr> "BLACK", "BLACK", "BLACK", "BLACK", "BLACK", "~
$ X_COORD_CD          <dbl> 986050, 1016802, 1048632, 1014493, 1009149, 99~
$ Y_COORD_CD          <dbl> 214231.0, 250581.0, 198262.0, 242565.0, 190104~
$ Latitude            <dbl> 40.75469, 40.85440, 40.71063, 40.83242, 40.688~
$ Longitude           <dbl> -73.99350, -73.88233, -73.76777, -73.89071, -7~
$ Lon_Lat             <chr> "POINT (-73.9935 40.754692)", "POINT (-73.8823~
```

I already see lots of nulls, empty strings, missing values, etc. Let's take a closer look at some of the categorical columns to see if there are a limited number of consistently entered values or if they were entered as free text, which might be too difficult to clean.

```
# break out all the unique values and counts
desc_counts <- lapply(incident_df[,
  c(
    "LOC_CLASSFCTN_DESC",
    "LOCATION_DESC",
    "PERP_RACE",
    "VIC_RACE",
    "LOC_OF_OCCUR_DESC",
```

```

    "VIC_SEX",
    "PERP_SEX"
)
], table)

print(desc_counts)

```

\$LOC\_CLASSFCTN\_DESC

	(null)	COMMERCIAL	DWELLING	HOUSING	OTHER
25596	2	208	243	460	59
PARKING LOT	PLAYGROUND	STREET	TRANSIT	VEHICLE	
15	41	1886	23	29	

\$LOCATION\_DESC

	(null)	ATM
14977	1711	1
BANK	BAR/NIGHT CLUB	BEAUTY/NAIL SALON
3	668	119
CANDY STORE	CHAIN STORE	CHECK CASH
7	7	1
CLOTHING BOUTIQUE	COMMERCIAL BLDG	DEPT STORE
14	304	9
DOCTOR/DENTIST	DRUG STORE	DRY CLEANER/LAUNDRY
1	14	32
FACTORY/WAREHOUSE	FAST FOOD	GAS STATION
8	130	74
GROCERY/BODEGA	GYM/FITNESS FACILITY	HOSPITAL
750	4	77
HOTEL/MOTEL	JEWELRY STORE	LIQUOR STORE
35	14	42
LOAN COMPANY	MULTI DWELL - APT BUILD	MULTI DWELL - PUBLIC HOUS
1	2964	5007
NONE	PHOTO/COPY STORE	PVT HOUSE
175	1	983
RESTAURANT/DINER	SCHOOL	SHOE STORE
212	1	10
SMALL MERCHANT	SOCIAL CLUB/POLICY LOCATI	STORAGE FACILITY
44	73	1
STORE UNCLASSIFIED	SUPERMARKET	TELECOMM. STORE
37	21	11

VARIETY STORE

11

VIDEO STORE

8

\$PERP\_RACE

	(null)
9310	1141
AMERICAN INDIAN/ALASKAN NATIVE	ASIAN / PACIFIC ISLANDER
2	169
BLACK	BLACK HISPANIC
11903	1392
UNKNOWN	WHITE
1837	298
WHITE HISPANIC	
2510	

\$VIC\_RACE

AMERICAN INDIAN/ALASKAN NATIVE	ASIAN / PACIFIC ISLANDER
11	440
BLACK	BLACK HISPANIC
20235	2795
UNKNOWN	WHITE
70	728
WHITE HISPANIC	
4283	

\$LOC\_OF\_OCCUR\_DESC

	INSIDE	OUTSIDE
25596	460	2506

\$VIC\_SEX

F	M	U
2760	25790	12

\$PERP\_SEX

(null)	F	M	U
9310	1141	444	16168
			1499

Everything seems to be consistently entered (no misspellings or variations.) But there is a

weird mix of “unknown”, “U”, and “null”. It will probably be best to recode empty values as “Unknown for consistency. There is something weird in a few columns too.

```
unique(incident_df$PERP_RACE)
table(incident_df$PERP_RACE)
```

1. 'BLACK'
2. '(null)'
3. ''
4. 'UNKNOWN'
5. 'WHITE HISPANIC'
6. 'BLACK HISPANIC'
7. 'ASIAN / PACIFIC ISLANDER'
8. 'WHITE'
9. 'AMERICAN INDIAN/ALASKAN NATIVE'

		(null)
	9310	1141
AMERICAN INDIAN/ALASKAN NATIVE		ASIAN / PACIFIC ISLANDER
	2	169
BLACK		BLACK HISPANIC
11903		1392
UNKNOWN		WHITE
1837		298
WHITE HISPANIC		
2510		

Oh, that’s annoying - there is an empty string '' as one of the largest groups, I guess the best option will be to categorize that as "UNKNOWN" as well. While I’m at it I’m going to make the date time columns a little more usable by separating out the date and time and converting them to the right type.

---

## Cleanup

```
# make a nicer datetime column
clean_incident_df <- incident_df %>%
  mutate(
```

```

    Date = as.POSIXct(
      paste(OCCUR_DATE, OCCUR_TIME),
      format = "%m/%d/%Y %H:%M:%S"
    )
  ) %>%
  rename(
    # rename some of the hardest to read columns
    In_Out = LOC_OF_OCCUR_DESC,
    Location_Category = LOC_CLASSFCTN_DESC,
    Location_details = LOCATION_DESC
  ) %>%
  select(
    Date, BORO, Location_Category, Location_details,
    In_Out, OCCUR_DATE, OCCUR_TIME,
    -JURISDICTION_CODE, -X_COORD_CD, -Y_COORD_CD,
    -Latitude, -Longitude, -Lon_Lat, -PRECINCT,
    everything()
  ) %>%
  mutate(
    # Replace specific values in PERP_RACE and VIC_RACE using recode
    PERP_RACE = recode(PERP_RACE,
      "ASIAN / PACIFIC ISLANDER" = "ASIAN_PAC_ISLAND",
      "AMERICAN INDIAN/ALASKAN NATIVE" = "AM_INDIAN/ALASKAN"
    ),
    VIC_RACE = recode(VIC_RACE,
      "ASIAN / PACIFIC ISLANDER" = "ASIAN_PAC_ISLAND",
      "AMERICAN INDIAN/ALASKAN NATIVE" = "AM_INDIAN/ALASKAN"
    ),
    # recode some of the empty value to the corresponding unknown option
    # I'm going to handle the nulls later
    PERP_RACE = ifelse(PERP_RACE == "", "UNKNOWN", PERP_RACE
  ),
    PERP_SEX = ifelse(PERP_SEX == "", "U", PERP_SEX)
  )

# check that I have the columns and order that I wanted
glimpse(clean_incident_df)

# check that we fixed the empty string values
unique(clean_incident_df$PERP_RACE)

# printing a df is a little uglier in some ways but prevents text

```

```
# overlap when there are lots of columns or long column names
print(tail(clean_incident_df))
```

Rows: 28,562

Columns: 22

```
$ Date <dtm> 2022-05-05 00:10:00, 2022-07-04 22:20:00, 201~
$ BORO <chr> "MANHATTAN", "BRONX", "QUEENS", "BRONX", "BROO~
$ Location_Category <chr> "COMMERCIAL", "STREET", "", "", "", "", "", ""~
$ Location_details <chr> "VIDEO STORE", "(null)", "", "", "", "MULTI DW~
$ In_Out <chr> "INSIDE", "OUTSIDE", "", "", "", "", "", "", ""~
$ OCCUR_DATE <chr> "05/05/2022", "07/04/2022", "05/27/2012", "09/~
$ OCCUR_TIME <chr> "00:10:00", "22:20:00", "19:35:00", "21:00:00"~
$ INCIDENT_KEY <int> 244608249, 247542571, 84967535, 202853370, 270~
$ PRECINCT <int> 14, 48, 103, 42, 83, 23, 113, 77, 48, 49, 73, ~
$ JURISDICTION_CODE <int> 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0~
$ STATISTICAL_MURDER_FLAG <chr> "true", "true", "false", "false", "false", "fa~
$ PERP_AGE_GROUP <chr> "25-44", "(null)", "", "25-44", "25-44", "", ""~
$ PERP_SEX <chr> "M", "(null)", "U", "M", "M", "U", "U", "U", ""~
$ PERP_RACE <chr> "BLACK", "(null)", "UNKNOWN", "UNKNOWN", "BLAC~
$ VIC_AGE_GROUP <chr> "25-44", "18-24", "18-24", "25-44", "25-44", ""~
$ VIC_SEX <chr> "M", "M", "M", "M", "M", "M", "M", "M", "M", ""~
$ VIC_RACE <chr> "BLACK", "BLACK", "BLACK", "BLACK", "BLACK", ""~
$ X_COORD_CD <dbl> 986050, 1016802, 1048632, 1014493, 1009149, 99~
$ Y_COORD_CD <dbl> 214231.0, 250581.0, 198262.0, 242565.0, 190104~
$ Latitude <dbl> 40.75469, 40.85440, 40.71063, 40.83242, 40.688~
$ Longitude <dbl> -73.99350, -73.88233, -73.76777, -73.89071, -7~
$ Lon_Lat <chr> "POINT (-73.9935 40.754692)", "POINT (-73.8823~
```

1. 'BLACK'
2. '(null)'
3. 'UNKNOWN'
4. 'WHITE HISPANIC'
5. 'BLACK HISPANIC'
6. 'ASIAN\_PAC\_ISLAND'
7. 'WHITE'
8. 'AM\_INDIAN/ALASKAN'

	Date	BORO	Location_Category	Location_details
28557	2023-07-02 21:40:00	BRONX	STREET	(null)
28558	2023-03-19 23:48:00	BRONX	COMMERCIAL	GROCERY/BODEGA
28559	2023-08-16 02:46:00	BRONX	STREET	(null)
28560	2023-06-27 12:27:00	BRONX	DWELLING	MULTI DWELL - APT BUILD

28561	2023-07-08 11:27:00	QUEENS	STREET	BEAUTY/NAIL SALON
28562	2023-07-24 23:38:00	MANHATTAN	HOUSING MULTI DWELL - PUBLIC HOUS	
	In_Out	OCCUR_DATE	OCCUR_TIME	INCIDENT_KEY PRECINCT JURISDICTION_CODE
28557	OUTSIDE	07/02/2023	21:40:00	270719378 46 0
28558	INSIDE	03/19/2023	23:48:00	265354835 47 0
28559	OUTSIDE	08/16/2023	02:46:00	272968931 41 0
28560	INSIDE	06/27/2023	12:27:00	270489846 41 0
28561	OUTSIDE	07/08/2023	11:27:00	271021661 102 0
28562	OUTSIDE	07/24/2023	23:38:00	271818283 28 2
	STATISTICAL_MURDER_FLAG	PERP_AGE_GROUP	PERP_SEX	PERP_RACE
28557		false	(null)	(null)
28558		true	18-24	M BLACK
28559		false	25-44	F BLACK
28560		true	25-44	M BLACK
28561		false	25-44	M WHITE HISPANIC
28562		false	(null)	(null)
	VIC_AGE_GROUP	VIC_SEX	VIC_RACE	X_COORD_CD Y_COORD_CD Latitude
28557	18-24	M	BLACK HISPANIC	1009601 247515 40.84601
28558	18-24	M	BLACK	1025687 268586 40.90378
28559	45-64	M	BLACK	1014639 240066 40.82555
28560	25-44	M	BLACK	1012221 238552 40.82140
28561	65+	M	ASIAN_PAC_ISLAND	1028856 192785 40.69572
28562	25-44	M	BLACK	997853 230889 40.80040
	Longitude	Lon_Lat		
28557	-73.90837	POINT (-73.908369 40.846012)		
28558	-73.85010	POINT (-73.850098 40.903785)		
28559	-73.89020	POINT (-73.890195 40.825549)		
28560	-73.89894	POINT (-73.898938 40.821404)		
28561	-73.83914	POINT (-73.839138 40.695717)		
28562	-73.95086	POINT (-73.950864 40.800405)		

I'm going to make a few different dataframes with different groups for eventual analysis and plotting. Things I'm going to start with

- Daily incidents over time to look for general trends
- Incidents by month and year
- Incidents by borough
- Incidents by month (not over time, so total incidents that occurred in each month summed over all years)
- Time and year data broken down by borough



```

# for plotting incidents over time
time_series_df <- clean_incident_df %>%
  mutate(simple_date = as.Date(OCCUR_DATE, format = "%m/%d/%Y")) %>%
  group_by(simple_date) %>%
# Add a new column that represents only the month and year
# This step may be unnecessary since I have a good date column
# but it's easier for me to understand
  summarise(total_by_day = n()) %>%
  mutate(month_year = floor_date(simple_date, "month"))

# for plotting overtime by month and year
df_aggregated <- time_series_df %>%
  mutate(year = format(simple_date, "%Y"),
         month = format(simple_date, "%m")) %>%
  group_by(year, month) %>%
  summarise(total_by_day = sum(total_by_day)) %>%
  ungroup()

# borough totals
total_by_borough <- clean_incident_df %>%
  group_by(BORO) %>%
  summarize(total_incidents = n())

# monthly borough totals
monthly_totals_by_borough <- clean_incident_df %>%
  mutate(month = floor_date(Date, "month")) %>%
  mutate(month = as.Date(month)) %>%
  group_by(BORO, month) %>%
  summarize(monthly_incidents = n()) %>%
  ungroup()

tail(time_series_df)
tail(df_aggregated)
tail(total_by_borough)
tail(monthly_totals_by_borough)

```

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

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

A tibble: 6 x 3

simple_date <date>	total_by_day <int>	month_year <date>
2023-12-22	8	2023-12-01
2023-12-23	4	2023-12-01
2023-12-24	5	2023-12-01
2023-12-26	6	2023-12-01
2023-12-27	1	2023-12-01
2023-12-29	3	2023-12-01

A tibble: 6 x 3

year <chr>	month <chr>	total_by_day <int>
2023	07	152
2023	08	108
2023	09	105
2023	10	99
2023	11	71
2023	12	83

A tibble: 5 x 2

BORO <chr>	total_incidents <int>
BRONX	8376
BROOKLYN	11346
MANHATTAN	3762
QUEENS	4271
STATEN ISLAND	807

A tibble: 6 x 3

BORO <chr>	month <date>	monthly_incidents <int>
STATEN ISLAND	2023-05-01	3
STATEN ISLAND	2023-06-01	8
STATEN ISLAND	2023-07-01	6
STATEN ISLAND	2023-08-01	3
STATEN ISLAND	2023-10-01	3
STATEN ISLAND	2023-11-01	2

That's looks pretty good. I'm going to make all my plots at once, so it will get a little messy looking, but that will be the easiest way to set some universal configurations (theme, size, etc.) Then we can use these to decide on further plotting or analysis or modeling to do.

---

## Visualization

```
options(repr.plot.width = 10, repr.plot.height = 7)
theme_set(theme_minimal())

# plot daily incidents
ggplot(time_series_df, aes(x = simple_date, y = total_by_day)) +
  geom_line(color = "skyblue") +
  labs(
    title = "Fig.1 - Daily Incidents",
    x = "Date",
    y = "Incident Count"
  )

# plot monthly incidents over time
ggplot(time_series_df, aes(x = month_year, y = total_by_day)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(
    title = "Fig. 2 - Total Occurrences by Month",
    x = "Year",
    y = "Total Occurrences") +
  scale_x_date(date_labels = "%Y", date_breaks = "1 year")

# plot yearly incidents
ggplot(time_series_df, aes(
  x = year(simple_date),
  y = total_by_day)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(
    title = "Fig. 3 - Total Occurrences by Year",
    x = "Year",
    y = "Total Occurrences")
```

```

# Plot occurrences by month (across all years)
ggplot(time_series_df, aes(
  x = month(simple_date, label = TRUE),
  y = total_by_day
)) +
  geom_bar(
    stat = "identity",
    fill = "skyblue"
  ) +
  labs(
    title = "Fig. 4 - Total Occurrences by Month",
    x = "Month",
    y = "Total Occurrences"
  )

# Plot total by borough
ggplot(total_by_borough, aes(
  x = BORO,
  y = total_incidents
)) +
  geom_bar(
    stat = "identity",
    fill = "skyblue"
  ) +
  labs(
    title = "Fig. 5 - Total by Borough",
    x = "Borough",
    y = "Total"
  )

# Assuming you already have the 'monthly_totals' dataframe
ggplot(monthly_totals_by_borough, aes(
  x = month,
  y = monthly_incidents,
  color = BORO
)) +
  # I wanted to try a line instead of a bar
  geom_line(linewidth = 1.2) +
  geom_point(size = 2) +
  labs(
    title = "Fig. 6 - Monthly Incidents by Borough",

```

```

    x = "Month",
    y = "Total Incidents"
  ) +
  scale_x_date(date_labels = "%b %Y", date_breaks = "6 month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

# Plot incidents by year for each borough with side-by-side bars
ggplot(monthly_totals_by_borough, aes(
  x = year(month),
  y = monthly_incidents,
  fill = BORO
)) +
  # Use dodge for side-by-side bars
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Fig. 7 - Yearly Incidents by Borough",
    x = "Year",
    y = "Total Incidents")

```

Fig.1 - Daily Incidents

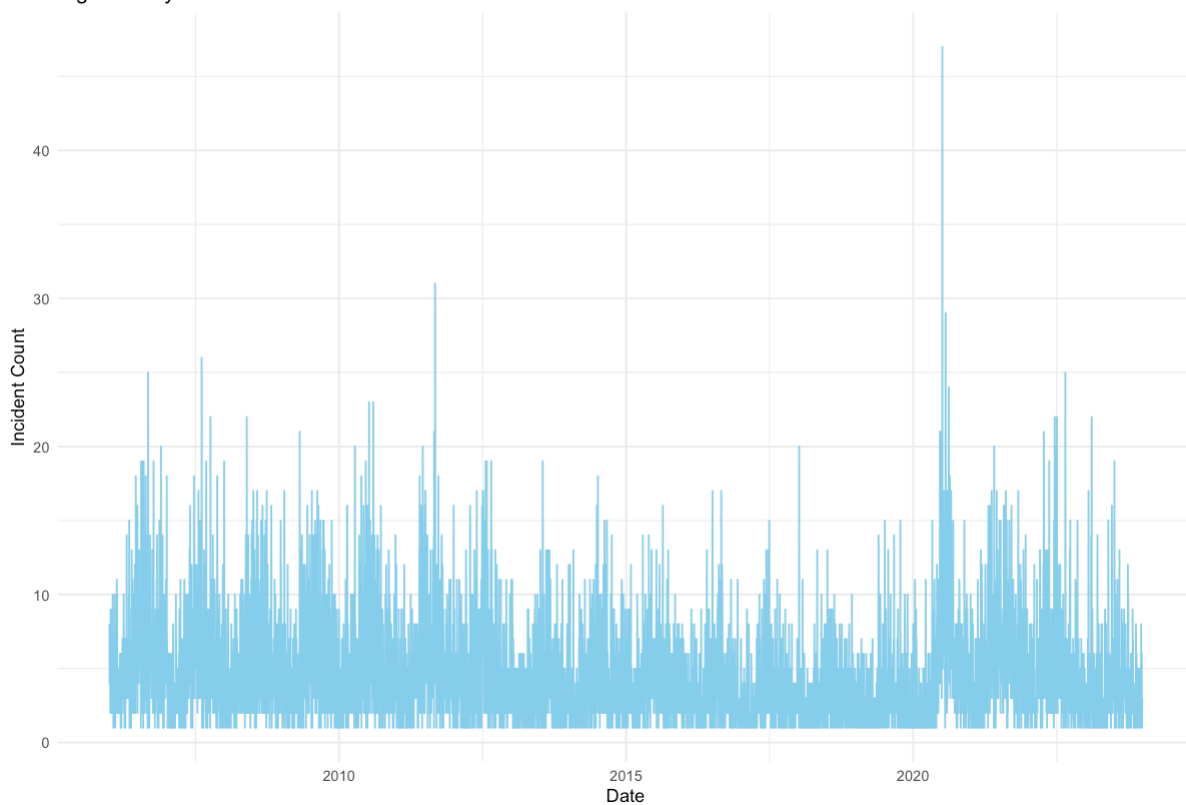


Fig. 2 - Total Occurrences by Month

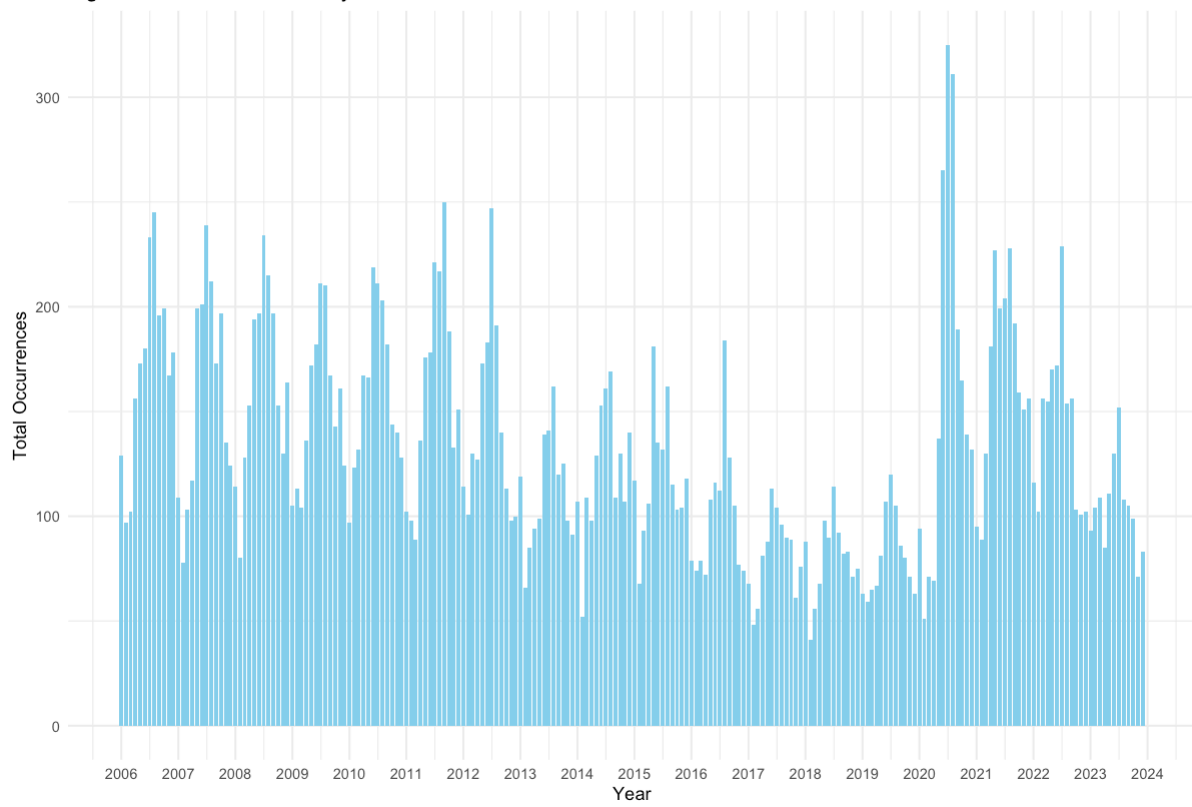


Fig. 3 - Total Occurrences by Year

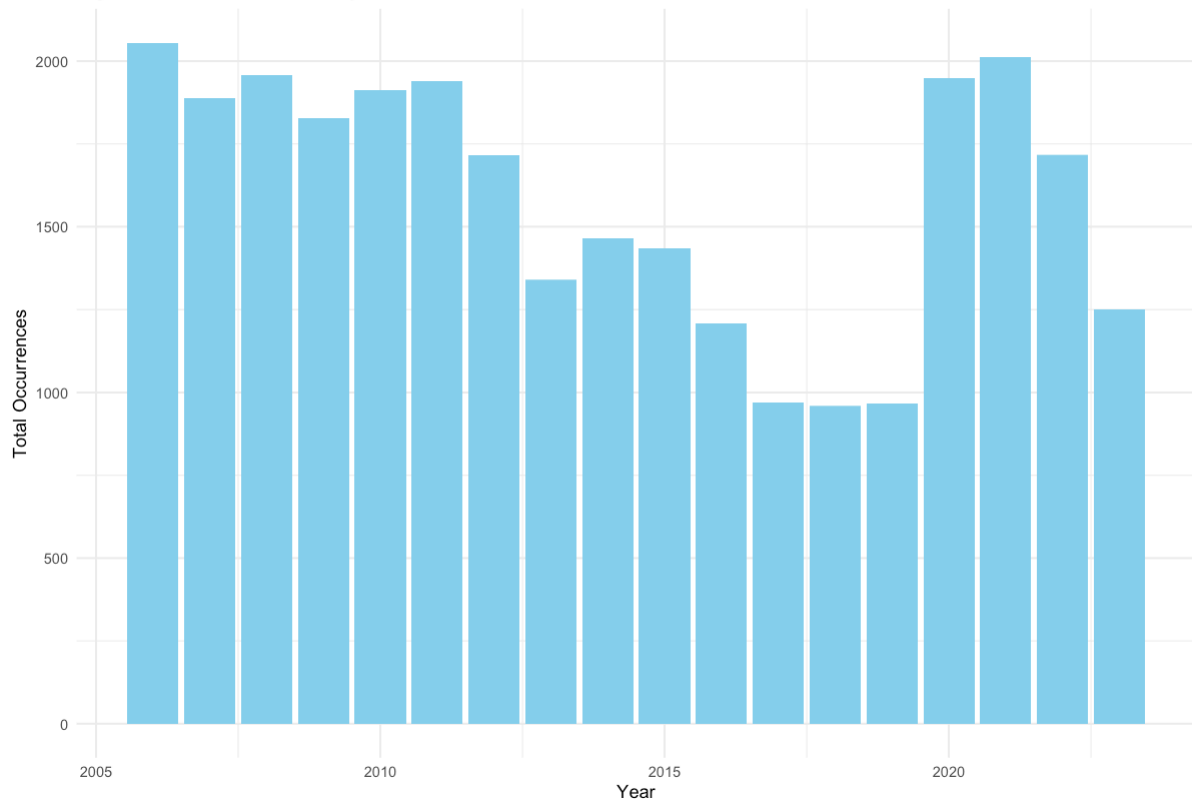


Fig. 4 - Total Occurrences by Month

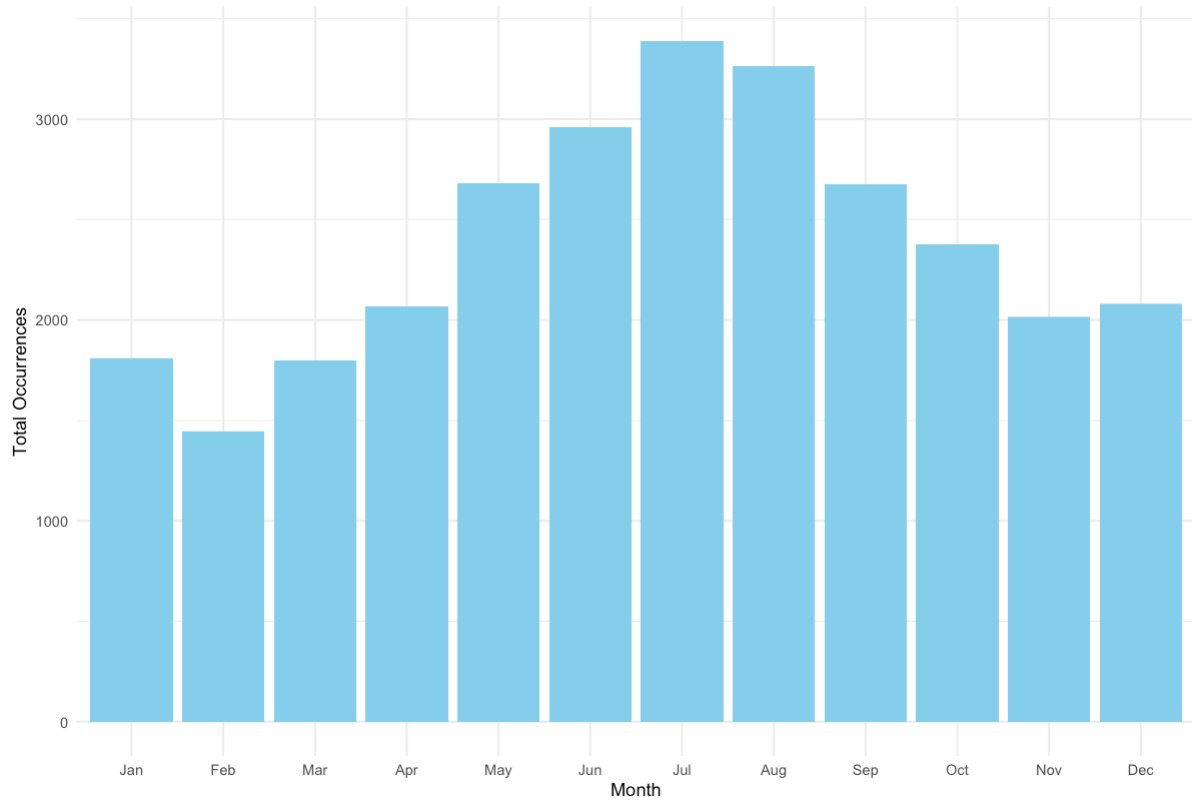




Fig. 5 - Total by Borough

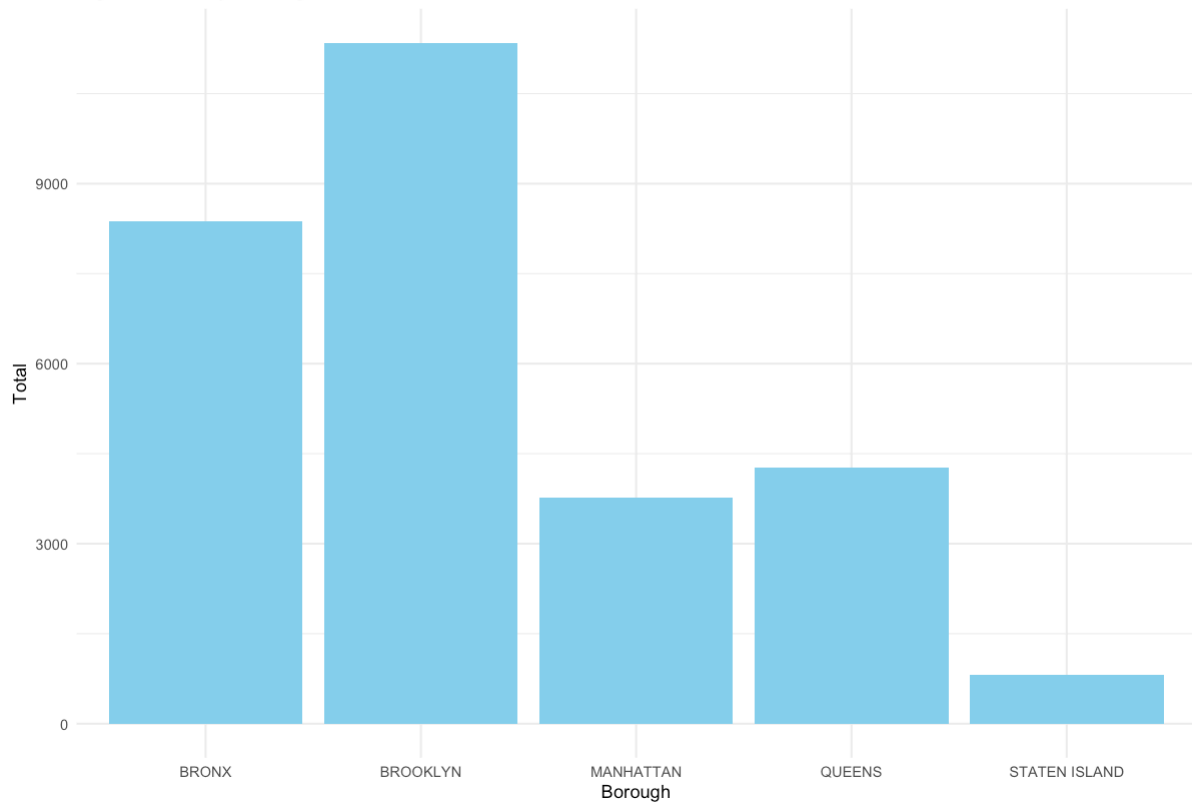


Fig. 6 - Monthly Incidents by Borough

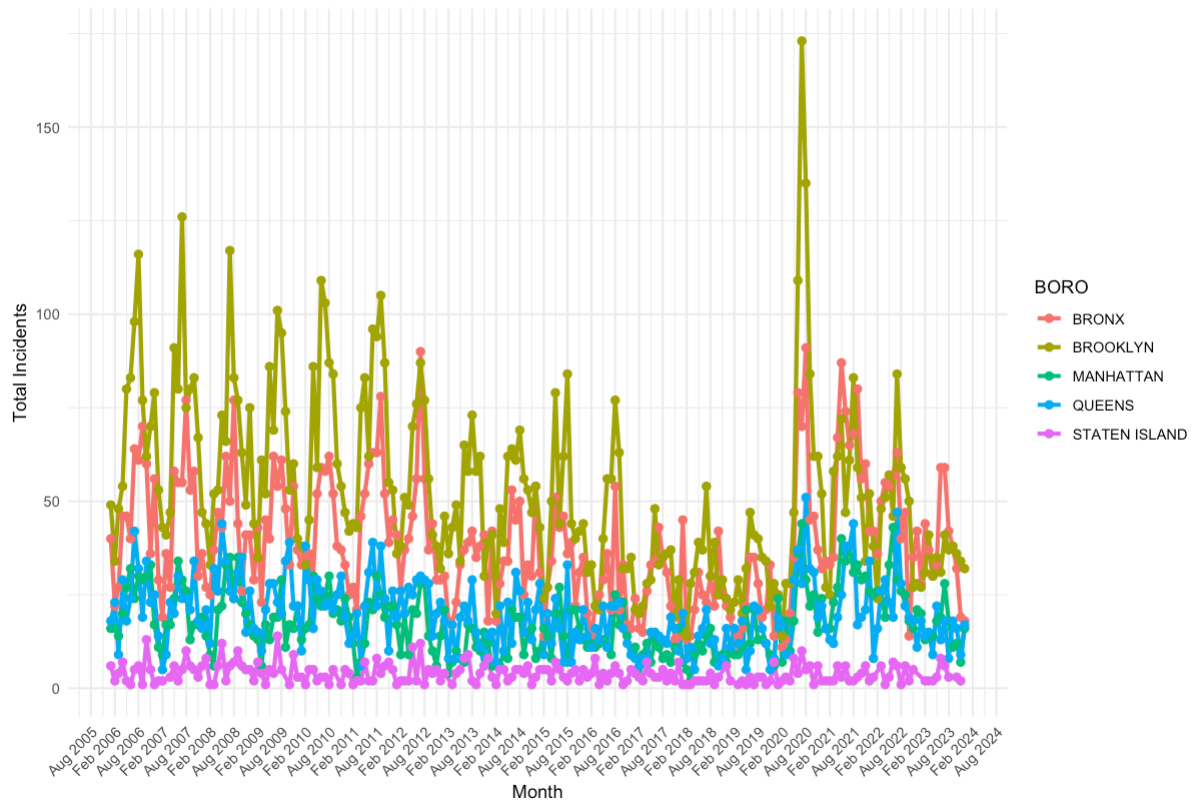
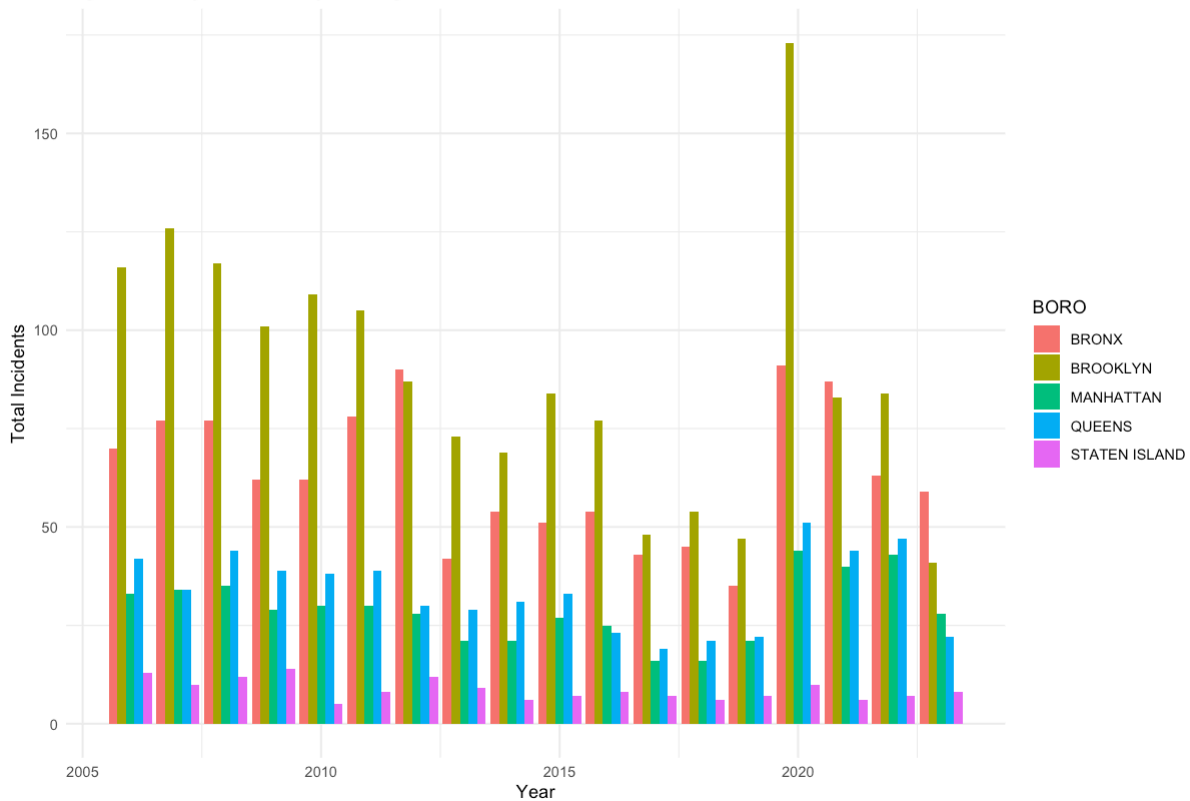


Fig. 7 - Yearly Incidents by Borough



- Fig 1 - Daily incidents - hard to see many trends since the data is so noisy, but it does look like it's generally periodic and there is a big spike around the first covid summer.
- Fig 2 - Total occurrences by month over time - clearer seasonality and a little easier to see the pre- and post-covid trends
- Fig 3 - Total occurrences by year - now we can see trends. Decreasing incidents starting in the early 20-teens and flattening out, before a big covid spike and almost back down to pre-covid levels.
- Fig 4 - Total occurrences by month - pretty strong visual trend towards higher incidents in the hottest months, which is a well studied phenomenon.
- Fig 5 - Total by borough - generally interesting, but would be more useful with the context of per capita and per area data for the boroughs.
- Fig 6/7 - Boroughs over time (final 2 plots) - interesting to see that not all boroughs follow the same trends over time, and that the first covid spike was driven heavily by increased in Brooklyn.

We could keep going with similar visuals (breakdown by race, gender, age group, etc. or relationships between group like victim age relative to perpetrator age) but I'll stop there. I'm going to focus on victim sex for the analysis and modeling component. I want to see how predictive of a victim being female some of the other attributes (perpetrator race and sex,

victim race). I'll start with a simple model of victim sex as predicted by perpetrator sex. To start will have to exclude the “unknowns” from victim sex and clean up some of the other factors.

---

## Modeling

```
# Recode "U" as NA for victim
clean_incident_df$VIC_SEX[clean_incident_df$VIC_SEX == "U"] <- NA

# create new df for modeling
#replace null/na in some of the factors with "unknown" for consistency
model_df <- clean_incident_df %>%
  mutate(
    PERP_SEX = ifelse(PERP_SEX == "(null)", "U", PERP_SEX),
    PERP_RACE = ifelse(PERP_RACE == "(null)", "UNKNOWN", PERP_RACE)
  )

# Convert all categorical variables to factors
model_df$VIC_SEX <- factor(model_df$VIC_SEX)
model_df$BORO <- factor(model_df$BORO)
model_df$VIC_RACE <- factor(model_df$VIC_RACE)
model_df$PERP_RACE <- factor(model_df$PERP_RACE)
model_df$PERP_SEX <- factor(model_df$PERP_SEX)

# Apply droplevels to all factor columns (to remove unused levels)
model_df <- model_df %>%
  mutate_if(is.factor, droplevels)

# Set "M" as the victim reference so that we model the odds of being "Female"
# Set the reference race for victims and perps as "white"
model_df$VIC_SEX <- relevel(model_df$VIC_SEX, ref = "M")
model_df$VIC_RACE <- relevel(model_df$VIC_RACE, ref = "WHITE")
model_df$PERP_RACE <- relevel(model_df$PERP_RACE, ref = "WHITE")

# Check the levels of the factor to confirm they are correct
levels(model_df$VIC_SEX)
levels(model_df$PERP_SEX)
levels(model_df$PERP_RACE)
```

1. 'M'
  2. 'F'
- 
1. 'F'
  2. 'M'
  3. 'U'
- 
1. 'WHITE'
  2. 'AM\_INDIAN/ALASKAN'
  3. 'ASIAN\_PAC\_ISLAND'
  4. 'BLACK'
  5. 'BLACK HISPANIC'
  6. 'UNKNOWN'
  7. 'WHITE HISPANIC'

```
# create a simple generalize linear model to predict odds of female victim
# based on perp sex
simple_model_vic_sex <- glm(VIC_SEX ~ PERP_SEX,
  family=binomial,
  data=model_df,
  na.action = na.exclude)

summary(simple_model_vic_sex)
```

Call:

```
glm(formula = VIC_SEX ~ PERP_SEX, family = binomial, data = model_df,
    na.action = na.exclude)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-1.5588	0.1254	-12.433	< 2e-16 ***
PERP_SEXM	-0.5463	0.1279	-4.272	1.94e-05 ***
PERP_SEXU	-0.9154	0.1300	-7.044	1.86e-12 ***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 18141 on 28549 degrees of freedom  
 Residual deviance: 18038 on 28547 degrees of freedom  
 (12 observations deleted due to missingness)  
 AIC: 18044

Number of Fisher Scoring iterations: 5

It looks like each perpetrator sex option has a statistically significant effect on the change in log-odds of the victim being female, so I'll pull out each of them and convert them to odds, so I'll pull them each out and convert them to odds.

```
print("Log-odds are")
print(coef(simple_model_vic_sex)[c("PERP_SEXM", "PERP_SEXU")])

#convert log-odds to odds and print
print("The odds relative to the female victim/female perpetrator baseline are")
print((exp(coef(simple_model_vic_sex)[c("PERP_SEXM", "PERP_SEXU")]))))

# Print odds for when the perpetrator is female
paste(
  "The odds of a victim being female when the perpetrator is female are",
  round(exp(coef(simple_model_vic_sex)["(Intercept)"]), 2)
)

# Print odds for when the perpetrator is male
paste0(
  "The odds of a victim being female when the perpetrator is male are ",
  round(exp(coef(simple_model_vic_sex)["PERP_SEXM"]), 2) * 100,
  "% of baseline"
)

# Print odds for when the perpetrator is unknown
paste0(
  "The odds of a victim being female when the perpetrator is unknown are ",
  round(exp(coef(simple_model_vic_sex)["PERP_SEXU"]), 2) * 100,
  "% of baseline"
)
```

```
[1] "Log-odds are"
      PERP_SEXM PERP_SEXU
-0.5463480 -0.9154259
[1] "The odds relative to the female victim/female perpetrator baseline are"
      PERP_SEXM PERP_SEXU
0.5790607 0.4003461
```

'The odds of a victim being female when the perpetrator is female are 0.21'

‘The odds of a victim being female when the perpetrator is male are 58% of baseline’

‘The odds of a victim being female when the perpetrator is unknown are 40% of baseline’

That’s the end of the analysis I’m comfortable with. Below I wanted to see what it would look like to do similar modeling with multiple predictor variables (perp sex, race, borough.) It looks like it worked, but it gets out of hand to interpret it pretty quickly so I just stopped and left it here as an interesting example of what else could be done.

```
model_vic_sex <- glm(VIC_SEX ~ BORO + VIC_RACE +  
  PERP_SEX + PERP_RACE,  
  family=binomial,  
  data=model_df,  
  na.action = na.exclude)  
  
print(levels(model_df$VIC_RACE))  
print(levels(model_df$PERP_RACE))  
print(levels(model_df$BORO))  
print(summary(model_vic_sex))
```

```
[1] "WHITE"           "AM_INDIAN/ALASKAN" "ASIAN_PAC_ISLAND"  
[4] "BLACK"           "BLACK HISPANIC"    "UNKNOWN"  
[7] "WHITE HISPANIC"  
[1] "WHITE"           "AM_INDIAN/ALASKAN" "ASIAN_PAC_ISLAND"  
[4] "BLACK"           "BLACK HISPANIC"    "UNKNOWN"  
[7] "WHITE HISPANIC"  
[1] "BRONX"           "BROOKLYN"          "MANHATTAN"         "QUEENS"  
[5] "STATEN ISLAND"
```

Call:

```
glm(formula = VIC_SEX ~ BORO + VIC_RACE + PERP_SEX + PERP_RACE,  
     family = binomial, data = model_df, na.action = na.exclude)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	-1.078413	0.206495	-5.222	1.77e-07	***
BOROBROOKLYN	0.157686	0.052155	3.023	0.002499	**
BOROMANHATTAN	0.108796	0.067349	1.615	0.106220	
BOROQUEENS	0.182218	0.064675	2.817	0.004841	**
BOROSTATEN ISLAND	0.227746	0.117056	1.946	0.051700	.
VIC_RACEAM_INDIAN/ALASKAN	-0.583073	1.057088	-0.552	0.581233	
VIC_RACEASIAN_PAC_ISLAND	-0.706628	0.199946	-3.534	0.000409	***
VIC_RACEBLACK	-0.628932	0.112511	-5.590	2.27e-08	***

VIC_RACEBLACK HISPANIC	-0.566739	0.128208	-4.420	9.85e-06	***
VIC_RACEUNKNOWN	-2.394956	1.014271	-2.361	0.018213	*
VIC_RACEWHITE HISPANIC	-0.300278	0.119115	-2.521	0.011705	*
PERP_SEXM	-0.533656	0.128639	-4.148	3.35e-05	***
PERP_SEXU	-0.406559	0.236166	-1.722	0.085160	.
PERP_RACEAM_INDIAN/ALASKAN	-8.691789	84.201452	-0.103	0.917784	
PERP_RACEASIAN_PAC_ISLAND	0.200856	0.282430	0.711	0.476980	
PERP_RACEBLACK	0.007634	0.174752	0.044	0.965155	
PERP_RACEBLACK HISPANIC	-0.317345	0.196338	-1.616	0.106025	
PERP_RACEUNKNOWN	-0.536431	0.261226	-2.054	0.040023	*
PERP_RACEWHITE HISPANIC	-0.253171	0.183560	-1.379	0.167824	

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 18141 on 28549 degrees of freedom  
 Residual deviance: 17939 on 28531 degrees of freedom  
 (12 observations deleted due to missingness)  
 AIC: 17977

Number of Fisher Scoring iterations: 9

---

## Potential Bias

There are several sources of potential bias in this report

- Collection bias - Some of the categorical variable options, such as the descriptions of locations or locations categories, are limited and some actual locations may be ambiguous. The racial categorization options are also somewhat limited compared to the true diversity of racial and ethnic backgrounds. The racial categorization is likely to have been chosen by someone other than the subject, so the selected race may not be accurate or reflect how the subject (victim or perpetrator) would self-identify.
- Analysis bias - There are likely many interesting insights in this data regarding race (relationship between race of perpetrator and victim, racial distribution of incidents throughout the boroughs, etc), I deliberately avoided any in-depth analysis of those categories as I believed it would be more complicated and nuanced than my current ability and available time would allow me to do well.



```
sessionInfo()
```

```
R version 4.4.1 (2024-06-14)
Platform: aarch64-apple-darwin20
Running under: macOS Sonoma 14.6.1
```

```
Matrix products: default
```

```
BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;
```

```
locale:
```

```
[1] C
```

```
time zone: America/Denver
```

```
tzcode source: internal
```

```
attached base packages:
```

```
[1] stats      graphics  grDevices  utils      datasets  methods    base
```

```
other attached packages:
```

```
[1] broom_1.0.6      lubridate_1.9.3 forcats_1.0.0    stringr_1.5.1
[5] dplyr_1.1.4      purrr_1.0.2      readr_2.1.5      tidyr_1.3.1
[9] tibble_3.2.1     ggplot2_3.5.1    tidyverse_2.0.0
```

```
loaded via a namespace (and not attached):
```

```
[1] gtable_0.3.5      jsonlite_1.8.8    compiler_4.4.1    crayon_1.5.3
[5] tidyselect_1.2.1  IRdisplay_1.1      scales_1.3.0      uuid_1.2-0
[9] fastmap_1.2.0     IRkernel_1.3.2    R6_2.5.1          labeling_0.4.3
[13] generics_0.1.3    backports_1.5.0    munsell_0.5.1     pillar_1.9.0
[17] tzdb_0.4.0        rlang_1.1.4        utf8_1.2.4        stringi_1.8.4
[21] repr_1.1.7        timechange_0.3.0   cli_3.6.3         withr_3.0.1
[25] magrittr_2.0.3     digest_0.6.36      grid_4.4.1        base64enc_0.1-3
[29] hms_1.1.3         pbdZMQ_0.3-11      lifecycle_1.0.4   vctrs_0.6.5
[33] evaluate_0.24.0    glue_1.7.0         farver_2.1.2      fansi_1.0.6
[37] colorspace_2.1-1  tools_4.4.1        pkgconfig_2.0.3   htmltools_0.5.8.1
```