NYPD Report Final

M.O.

2024-08-06

Data Import

1. Packages used;

```
if (!require("hms")) install.packages("hms")
library(tidyverse)
library(lubridate)
library(hms)
```

2. Import data from NYC Incident Report website, read the csv into `raw_data` variable.

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

raw_data <- read.csv(url_in) %>%
  mutate(date = mdy(OCCUR_DATE),
         year = year(date),
         time = parse_time(OCCUR_TIME),
         boro = BORO,
         precinct = PRECINCT,
         murder_flag = STATISTICAL_MURDER_FLAG) %>%
  select(year, time, boro, precinct, murder_flag)

summary(raw_data)
```

```
##       year          time             boro              precinct
##  Min.   :2006   Length:28562      Length:28562       Min.   :  1.0
##  1st Qu.:2009   Class1:hms        Class :character   1st Qu.: 44.0
##  Median :2013   Class2:difftime   Mode  :character   Median : 67.0
##  Mean   :2014   Mode  :numeric                       Mean   : 65.5
##  3rd Qu.:2019                                       3rd Qu.: 81.0
##  Max.   :2023                                       Max.   :123.0
##  murder_flag
##  Length:28562
##  Class :character
##  Mode  :character
##
##
##
```

EDA

1. Group by precinct, year, murder_flag and count up shooting incidents.

```
trend_df <- raw_data %>%
  group_by(precinct, year, murder_flag) %>%
  count() %>%
  ungroup()

trend_df
```

```
## # A tibble: 2,249 x 4
##    precinct  year murder_flag     n
##       <int> <dbl> <chr>       <int>
##  1        1  2007 false           1
##  2        1  2008 false           1
##  3        1  2009 true            1
##  4        1  2010 false           4
##  5        1  2010 true            4
##  6        1  2012 false           2
##  7        1  2013 false           1
##  8        1  2015 true            1
##  9        1  2017 false           2
## 10        1  2019 false           2
## # i 2,239 more rows
```

2. Pivot wider to separate murder and non-murder cases

```
trend_df2 <- trend_df %>%
  pivot_wider(names_from = murder_flag, values_from = n) %>%
  rename(cases = false,
         deaths = true) %>%
  replace_na(list(cases = 0, deaths = 0))

trend_df2
```

```
## # A tibble: 1,274 x 4
##    precinct  year cases deaths
##       <int> <dbl> <int>  <int>
##  1        1  2007     1      0
##  2        1  2008     1      0
##  3        1  2009     0      1
##  4        1  2010     4      4
##  5        1  2012     2      0
##  6        1  2013     1      0
##  7        1  2015     0      1
##  8        1  2017     2      0
##  9        1  2019     2      0
## 10        1  2020     2      1
## # i 1,264 more rows
```

3. Import precinct population data

```r
#Importing 2020 census data
pop_url <- "https://raw.githubusercontent.com/jkeefe/census-by-precincts/master/data/nyc/nyc_block_preci
pop_csv <- read.csv(pop_url) %>%
  select(precinct, P1_001N) %>%
  group_by(precinct) %>%
  mutate(pop_2020 = sum(P1_001N)) %>%
  select(precinct, pop_2020) %>%
  unique() %>%
  filter(!pop_2020==0)


#Importing 2010 census data
url_2010 <- "https://s3.amazonaws.com/media.johnkeefe.net/nypd-data/nyc_2010pop_2020precincts.csv"

pop2010_csv <- read.csv(url_2010) %>%
  select(precinct_2020, P0010001) %>%
  rename(precinct = precinct_2020,
         pop_2010 = P0010001) %>%
  unique()
```

4. Interpolate population for other years based on the difference between the 2020 and 2010 census data.

```r
precinct_pop <- pop_csv %>%
  left_join(pop2010_csv, by="precinct") %>%
  mutate(`2023` = pop_2020 - 3 * (pop_2020 * 0.024),
         `2022` = pop_2020 - 2 * (pop_2020 * 0.024),
         `2021` = pop_2020 - (pop_2020 * 0.024),
         `2020` = pop_2020,
         `2019` = pop_2020 - (pop_2020 - pop_2010)/10,
         `2018` = pop_2020 - 2 * (pop_2020 - pop_2010)/10,
         `2017` = pop_2020 - 3 * (pop_2020 - pop_2010)/10,
         `2016` = pop_2020 - 4 * (pop_2020 - pop_2010)/10,
         `2015` = pop_2020 - 5 * (pop_2020 - pop_2010)/10,
         `2014` = pop_2020 - 6 * (pop_2020 - pop_2010)/10,
         `2013` = pop_2020 - 7 * (pop_2020 - pop_2010)/10,
         `2012` = pop_2020 - 8 * (pop_2020 - pop_2010)/10,
         `2011` = pop_2020 - 9 * (pop_2020 - pop_2010)/10,
         `2010` = pop_2010,
         `2009` = pop_2010 - (pop_2010 * 0.031),
         `2008` = pop_2010 - 2 * (pop_2010 * 0.0031),
         `2007` = pop_2010 - 3 * (pop_2010 * 0.0031),
         `2006` = pop_2010 - 4 * (pop_2010 * 0.0031)) %>%
  select(-c(pop_2020, pop_2010)) %>%
  pivot_longer(!precinct, names_to = "year", values_to = "pop") %>%
  mutate(year = as.double(year))

precinct_pop$pop <- round(precinct_pop$pop, digits = 0)
head(precinct_pop)
```

```
## # A tibble: 6 x 3
## # Groups:   precinct [1]
##   precinct  year    pop
##      <int> <dbl>  <dbl>
```

```
## 1       43  2023 174478
## 2       43  2022 178990
## 3       43  2021 183503
## 4       43  2020 188015
## 5       43  2019 186426
## 6       43  2018 184836
```

5. Left join the `trend_df2` and `precinct_pop` to calculate injuries and deaths rates of each precinct.

```
trend_df_w_pop <- trend_df2 %>%
  left_join(precinct_pop, by=c("precinct" = "precinct", "year"="year")) %>%
  mutate(case_rate = cases / pop * 100000,
         death_rate = deaths / pop * 100000,
         precinct = as.factor(precinct))
trend_df_w_pop
```

```
## # A tibble: 1,274 x 7
##    precinct  year cases deaths   pop case_rate death_rate
##    <fct>    <dbl> <int>  <int> <dbl>     <dbl>      <dbl>
##  1 1         2007     1      0 66059      1.51       0
##  2 1         2008     1      0 66266      1.51       0
##  3 1         2009     0      1 64612      0          1.55
##  4 1         2010     4      4 66679      6.00       6.00
##  5 1         2012     2      0 70303      2.84       0
##  6 1         2013     1      0 72115      1.39       0
##  7 1         2015     0      1 75739      0          1.32
##  8 1         2017     2      0 79363      2.52       0
##  9 1         2019     2      0 82987      2.41       0
## 10 1         2020     2      1 84799      2.36       1.18
## # i 1,264 more rows
```

6. Find Max murder rate and non-murder rate

```
mean_df <- trend_df_w_pop %>%
  select(precinct, death_rate, case_rate) %>%
  group_by(precinct) %>%
  mutate(mean_death = mean(death_rate),
         mean_case = mean(case_rate)) %>%
  ungroup() %>%
  select(precinct, mean_death, mean_case) %>%
  unique()
```

7. Precincts with the highest murder rate

```
mean_df %>% slice_max(mean_death, n=10)
```

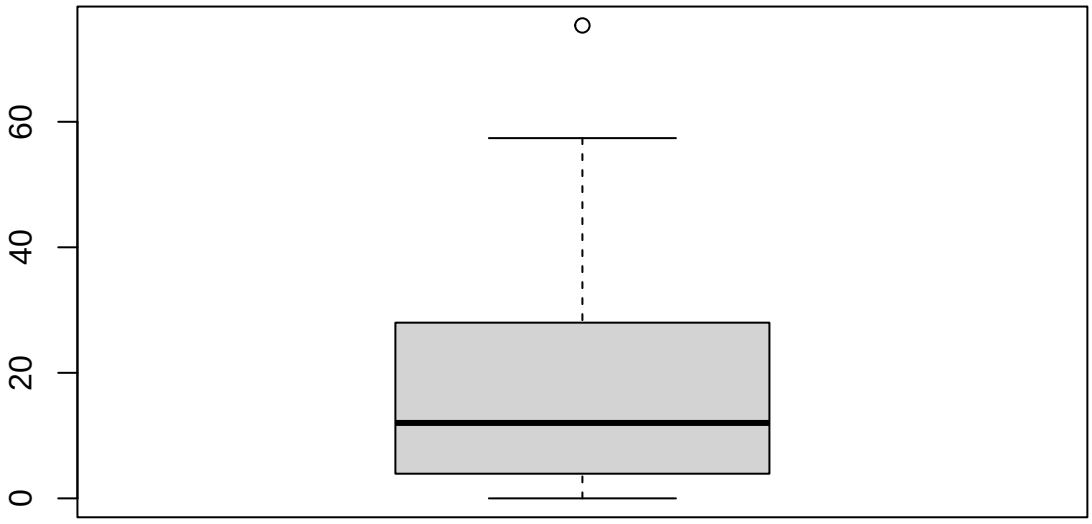
```
## # A tibble: 10 x 3
##    precinct mean_death mean_case
##    <fct>         <dbl>     <dbl>
##  1 73            17.1       75.3
##  2 81            13.5       57.4
```

```
##  3 79            11.5       49.8
##  4 42            11.3       47.0
##  5 48            10.8       44.3
##  6 41            10.5       44.4
##  7 77            10.4       38.3
##  8 25             9.75      47.1
##  9 40             9.72      46.0
## 10 75             9.68      38.4
```

The precincts that has the highest rate of deaths on average are 73, 81, 79, 42, 48, 41, 77, 25, 40, 75. There's an outlier in precinct 22.

8. Handling an outlier in precinct 22.

```r
mean_df$mean_case[mean_df$precinct == 22] <- 0
boxplot(mean_df$mean_case)
```



Visualization 1

1. Visualize the number of incidents in 2023 by precinct and year

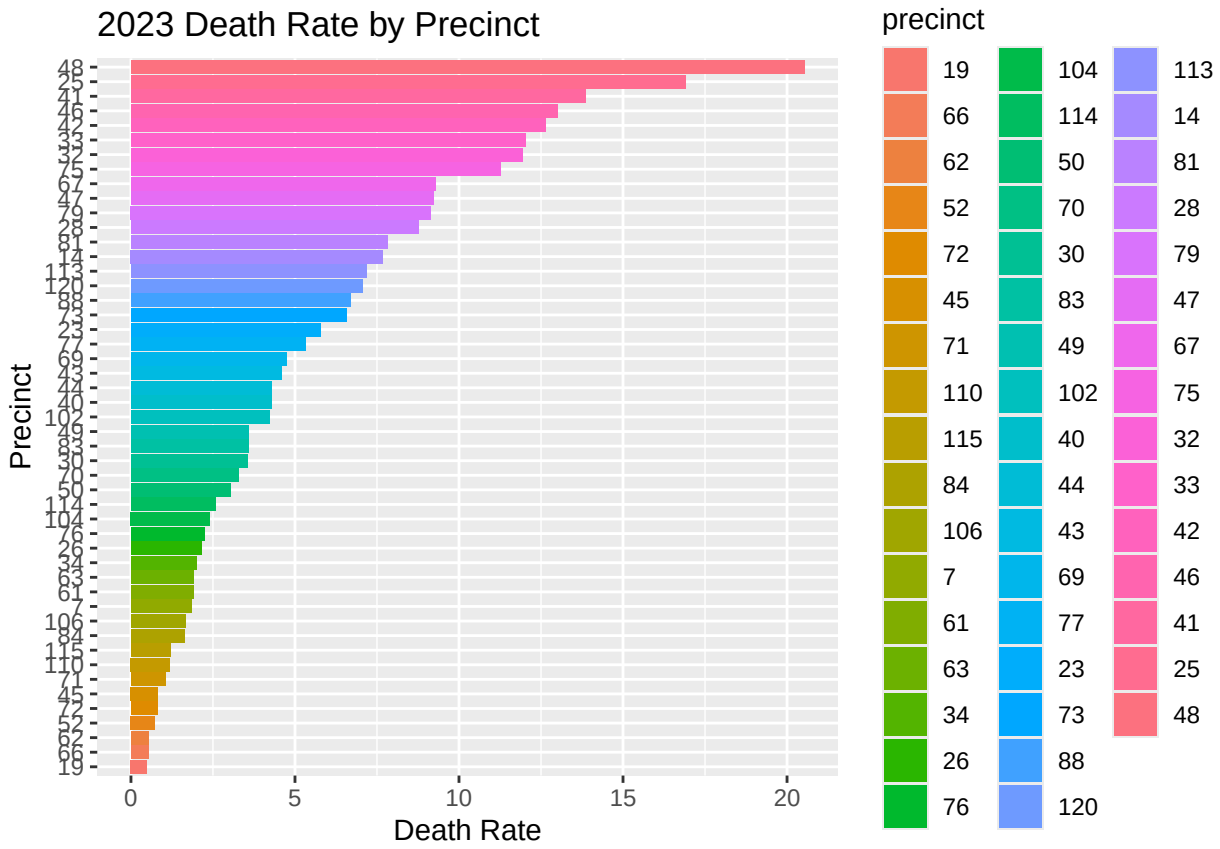
```r
# Extract 2023 data
total2023 <- trend_df_w_pop %>%
  filter(year == 2023, death_rate > 0)

# Reordering the data by n in a decreasing order
total2023 <- total2023[order(total2023$death_rate, decreasing = TRUE),]

total2023$precinct <- factor(total2023$precinct,
                             levels = total2023$precinct[order(total2023$death_rate)])


ggplot(data=total2023, aes(x=death_rate, y=precinct, group=precinct))+
  geom_col(aes(fill=precinct)) +
```

```
    labs(title="2023 Death Rate by Precinct",
       x="Death Rate", y="Precinct")
```



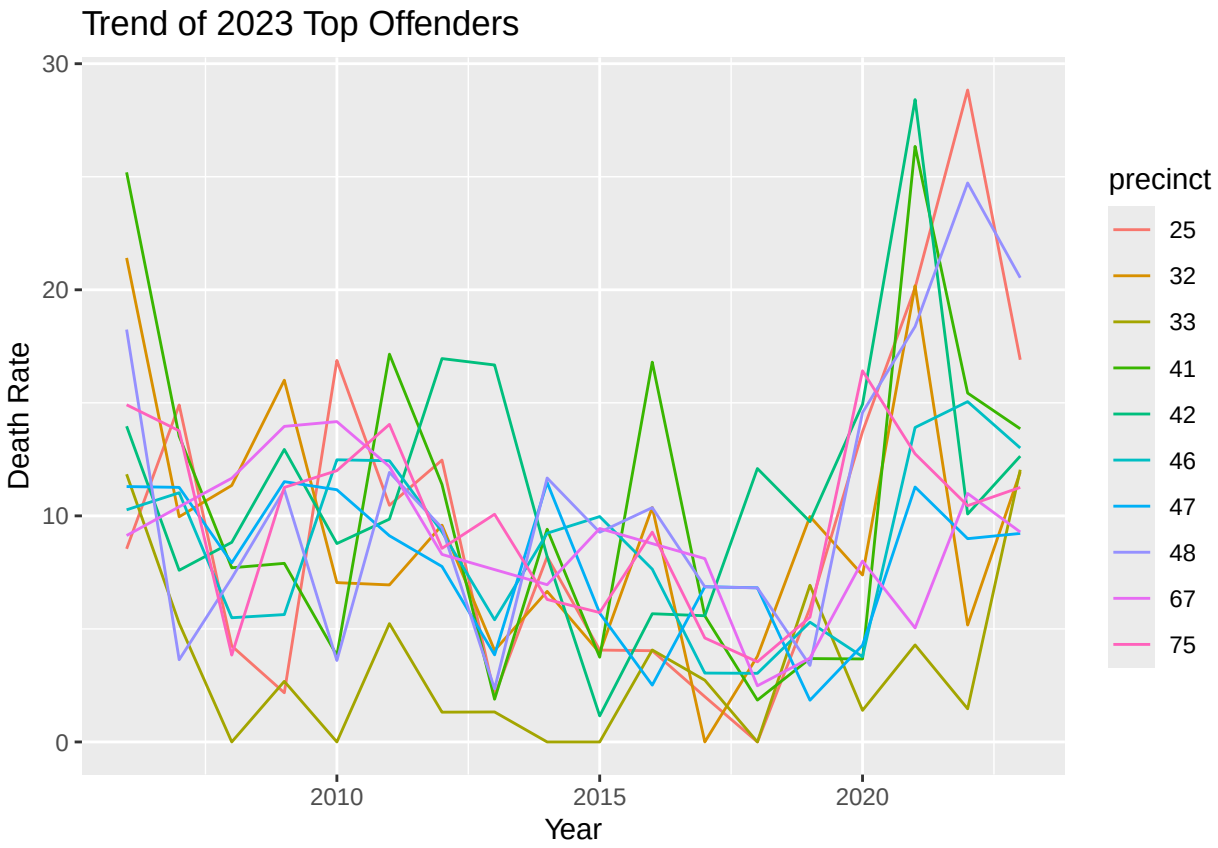
The top 10 precincts with the highest murder rate in 2023 are 48, 25, 41, 46, 42, 33, 32, 75, 67, 47.

Visualization 2

1. Visualize the trend of 2023 top offenders

```
top_offenders_2023 <- trend_df_w_pop %>%
  filter(precinct %in% c("48", "25", "41", "46", "42", "33", "32", "75", "67", "47"))


ggplot(data=top_offenders_2023, aes(x=year, y=death_rate))+
  geom_line(aes(colour = precinct))+
  labs(title="Trend of 2023 Top Offenders", x="Year", y="Death Rate")
```



While many precincts show downward trend in murder rate, there's a couple of precincts with increasing trend.

2. Find the precincts with increasing trend.

```
two_year_df <- trend_df_w_pop %>%
  filter(year %in% c(2022, 2023)) %>%
  select(precinct, death_rate, year) %>%
  pivot_wider(names_from = year, values_from = death_rate) %>%
  mutate(changes = `2023`-`2022`)

two_year_df %>%
  filter(precinct %in% c("48", "25", "41", "46", "42", "33", "32", "75", "67", "47")) %>%
  filter(changes > 3)
```

```
## # A tibble: 2 x 4
##   precinct `2022` `2023` changes
##   <fct>     <dbl>  <dbl>   <dbl>
## 1 32         5.17   11.9    6.77
## 2 33         1.47   12.0   10.6
```

Modeling the Data

1. Data preparation for model

```r
precinct_totals <- trend_df_w_pop %>% group_by(precinct) %>%
  summarize(deaths = max(deaths),
            cases = max(cases),
            population=max(pop),
            cases_per_100k = 100000 * cases/population,
            deaths_per_100k = 100000* deaths/population)

#Handling outlier
precinct_totals$cases_per_100k[precinct_totals$precinct == 22] <- 0

precinct_totals
```

```
## # A tibble: 77 x 6
##    precinct deaths cases population cases_per_100k deaths_per_100k
##    <fct>     <int> <int>      <dbl>          <dbl>           <dbl>
##  1 1             4     4      84799           4.72            4.72
##  2 5             4    11      52568          20.9             7.61
##  3 6             3     5      64643           7.73            4.64
##  4 7             3    13      57985          22.4             5.17
##  5 9             5     9      76443          11.8             6.54
##  6 10            6    11      65570          16.8             9.15
##  7 13            4     6     100050           6.00            4.00
##  8 14            6    18      28050          64.2            21.4
##  9 17            1     3      88343           3.40            1.13
## 10 18            2     6      67528           8.89            2.96
## # i 67 more rows
```

2. Create a linear model

```r
mod <- lm(deaths_per_100k ~ cases_per_100k, data=precinct_totals)

summary(mod)
```

```
##
## Call:
## lm(formula = deaths_per_100k ~ cases_per_100k, data = precinct_totals)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4762 -1.3495 -0.4466  1.1983  9.3589
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.08025    0.38422   2.811  0.00629 **
## cases_per_100k  0.28444    0.01009  28.178  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.181 on 75 degrees of freedom
## Multiple R-squared:  0.9137, Adjusted R-squared:  0.9125
## F-statistic:   794 on 1 and 75 DF,  p-value: < 2.2e-16
```

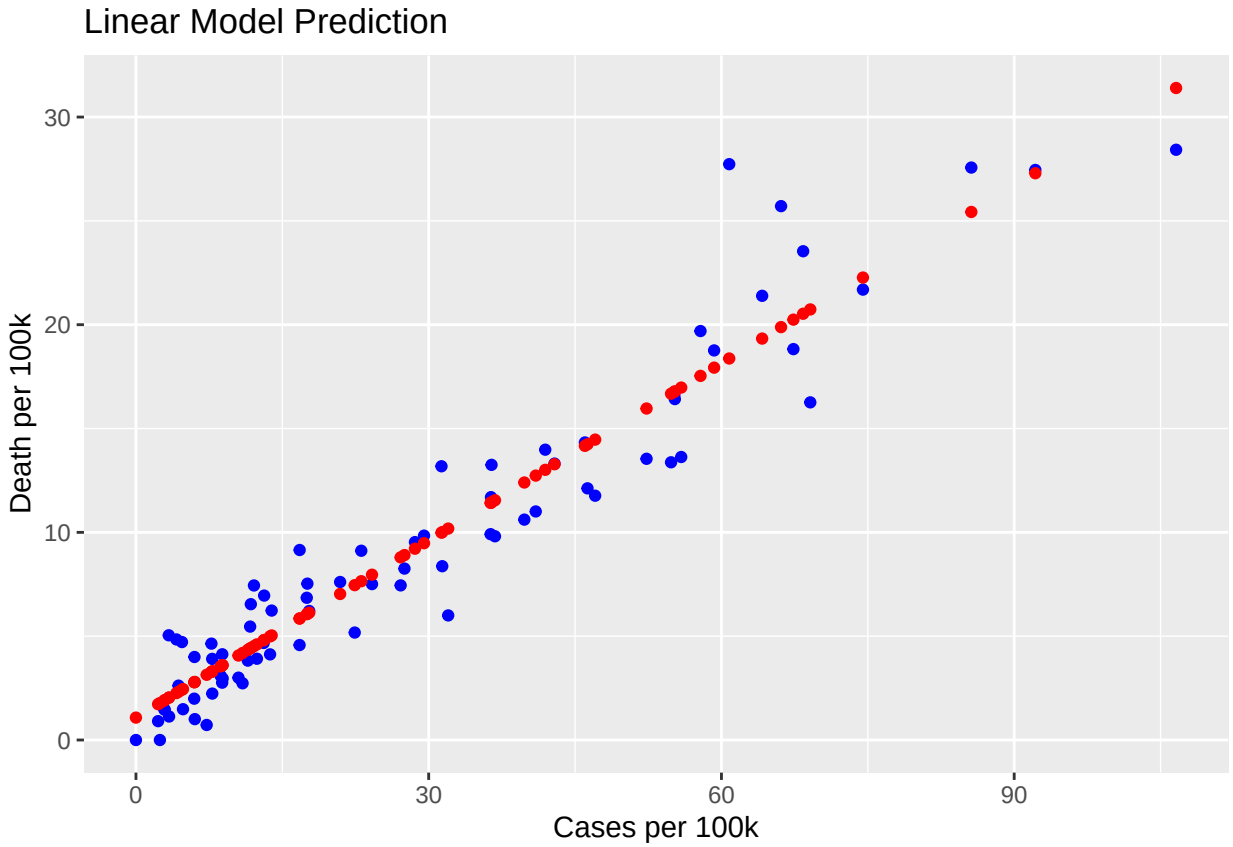
3. Add prediction to the data and visualize it.

```
tot_w_pred <- precinct_totals %>%
  ungroup() %>%
  mutate(pred = predict(mod))%>%
  select(cases_per_100k, deaths_per_100k, pred, everything())

tot_w_pred
```

```
## # A tibble: 77 x 7
##    cases_per_100k deaths_per_100k  pred precinct deaths cases population
##             <dbl>           <dbl> <dbl> <fct>     <int> <int>      <dbl>
##  1           4.72            4.72  2.42 1             4     4      84799
##  2          20.9             7.61  7.03 5             4    11      52568
##  3           7.73            4.64  3.28 6             3     5      64643
##  4          22.4             5.17  7.46 7             3    13      57985
##  5          11.8             6.54  4.43 9             5     9      76443
##  6          16.8             9.15  5.85 10            6    11      65570
##  7           6.00            4.00  2.79 13            4     6     100050
##  8          64.2            21.4  19.3  14            6    18      28050
##  9           3.40            1.13  2.05 17            1     3      88343
## 10           8.89            2.96  3.61 18            2     6      67528
## # i 67 more rows
```

```
tot_w_pred %>%
  ggplot() +
  geom_point(aes(x=cases_per_100k, y=deaths_per_100k), color="blue") +
  geom_point(aes(x=cases_per_100k, y=pred), color="red")+
  labs(title = "Linear Model Prediction", x="Cases per 100k", y="Death per 100k")
```



Summary

This analysis explored the relationship between murder and non-murder rate of shooting incidents using a linear model. The model suggests a strong linear correlation between the variables with the adjusted R-squared value of 0.9125.

While the analysis provide important insights into high-crime precincts, there are two biases involved in my data analysis.

First, there is a potential for data transformation bias. It's important to note that the yearly population of each precinct was estimated using interpolation based on two census data points in 2010 and 2020. Additionally, NYC's population declined from 8.8 million to 8.26 million between 2020 and 2023. This change rate was used across the board to estimate precinct populations for 2021 and later years. Furthermore, an annual change of 0.31%, obtained from an online source, was used to estimate precinct populations for 2006 to 2009. Therefore, the actual population figures may be different, potentially affecting the accuracy of the calculated case rate and death rates.

Second, there is a potential for outlier handling bias. Precinct 22, which covers Central Park, exhibits an outlier in its calculated case rate. This is because the precinct's population was only 129 in 2020. When one injury case was reported, dividing it by 129 and multiplying it by 100,000 resulted in an case rate of 775! This significantly deviates from the case rates of other precincts. A single case does not substantially impact the ultimate goal of reducing murder rates, I chose to address this outlier by setting its value to zero.

As a next step, I recommend analyzing crime patterns within these precincts, including time of day, day of week, and location-specific data to identify hotspots. Additionally, benchmarking high-crime precincts against lower-crime precincts would be beneficial for identifying potential best practices.