NYPD Shooting Incident

K. K.

2023-12-11

Load in data and select subset

The data collected by the NYPD on shooting incidents in New York City between 2006 and 2022 is first downloaded into R. A subset of the columns are selected for further investigation and renamed for easier reference in the code. The date is also converted into a date object from a string for better handling.

```
# load data from online
shoot_inci = read_csv('https://data.cityofnewyork.us/api/views/833y-fsy8/rows.csv?accessType=DOWNLOAD')
```

```
## Rows: 27312 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.
```

```
# choosing columns of personal interest
shoot_inci = select(shoot_inci,
                    c(OCCUR_DATE,OCCUR_TIME,BORO,)) %>%
  mutate(OCCUR_DATE = mdy(OCCUR_DATE)) %>%
  # change column titles to be easier to type
  rename(date = OCCUR_DATE,
         time = OCCUR_TIME,
         boro = BORO)

print(shoot_inci)
```

```
## # A tibble: 27,312 x 3
##    date       time   boro
##    <date>     <time> <chr>
##  1 2021-05-27 21:30  QUEENS
##  2 2014-06-27 17:40  BRONX
##  3 2015-11-21 03:56  QUEENS
##  4 2015-10-09 18:30  BRONX
##  5 2009-02-19 22:58  BRONX
##  6 2020-10-21 21:36  BROOKLYN
##  7 2012-06-17 22:47  QUEENS
##  8 2010-03-08 19:41  BROOKLYN
##  9 2012-02-05 05:45  QUEENS
## 10 2012-08-26 01:10  QUEENS
## # i 27,302 more rows
```

Analyze data

After browsing the data I began to wonder what the relationship is between the number of shooting incidents and the time of day. To investigate this, first the time is converted into a lubridate object. This allows dividing the incidents into 24 groups by the hour of the day. The count function was then used to determine the number of incidents that occurred during each hour within each borough.

```r
# is there a difference in number of incidents at different times of day?
inci_by_hour = shoot_inci %>%
  #group by hour of the day
  mutate(hour = lubridate::hour(time) %% 24) %>%
  group_by(boro, hour) %>%
  summarize(count = n()) %>%
  ungroup()
```

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

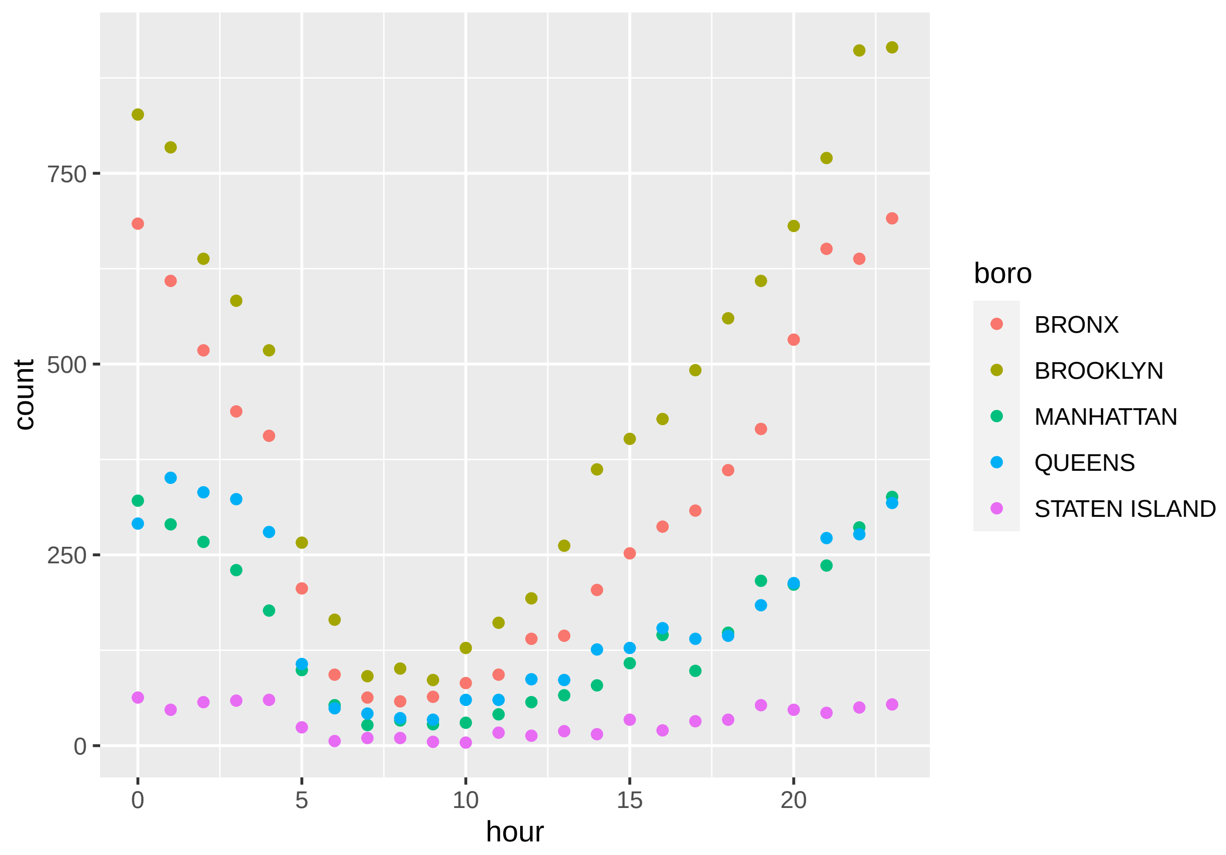
```r
print(inci_by_hour)
```

```
## # A tibble: 120 x 3
##    boro   hour count
##    <chr> <dbl> <int>
##  1 BRONX     0   684
##  2 BRONX     1   609
##  3 BRONX     2   518
##  4 BRONX     3   438
##  5 BRONX     4   406
##  6 BRONX     5   206
##  7 BRONX     6    93
##  8 BRONX     7    63
##  9 BRONX     8    58
## 10 BRONX     9    64
## # i 110 more rows
```

Visualize analysis

The best way to get a quick idea of a relationship is to make a graph.

```r
# number of incidents in each borough by hour of the day
ggplot(inci_by_hour, aes(x=hour,y=count, color=boro)) +
  geom_point()
```



The incidents graphed against hour of the day suggests a parabolic relationship.

Model incidents by hour for each borough

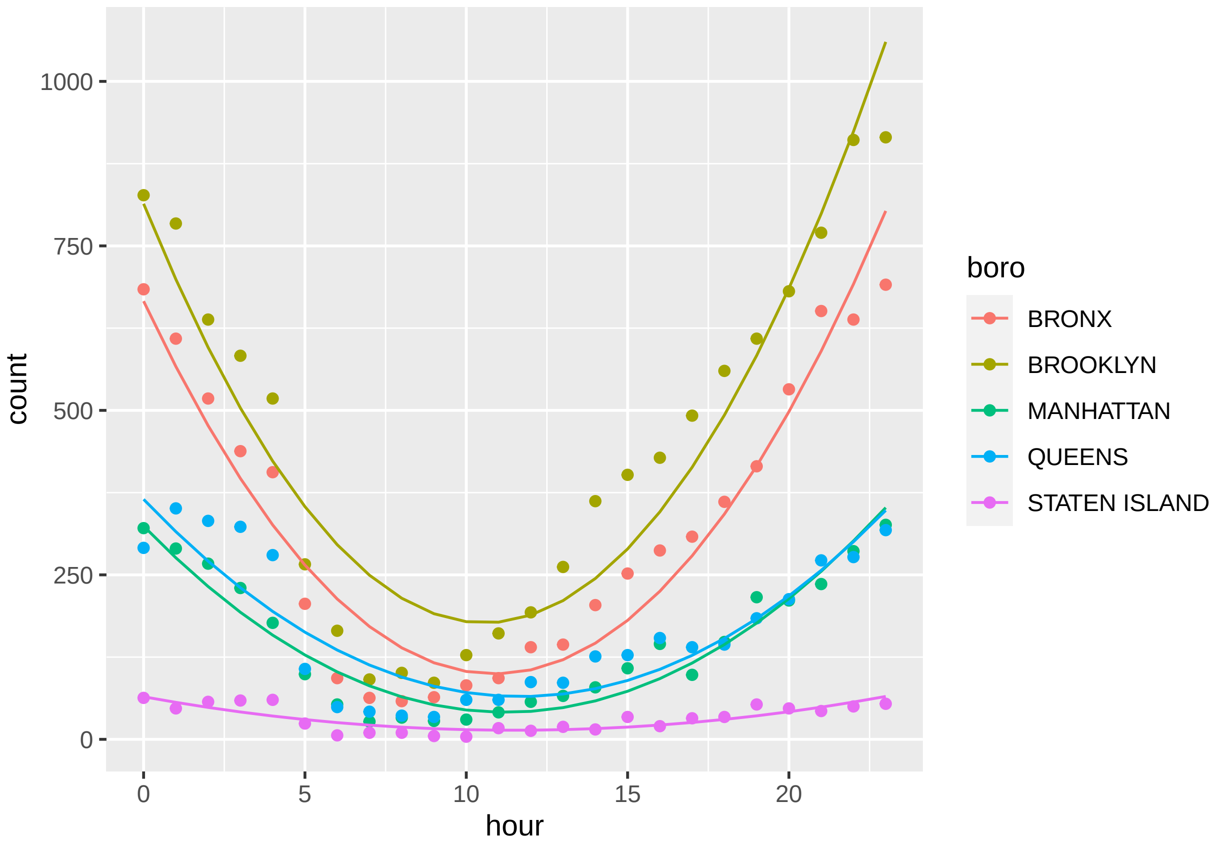
We can check how parabolic the relationship is by building quadratic models for each borough.

```r
inci_by_hour = inci_by_hour %>%
  #get hour^2 for quadratic model
  mutate(hour2 = hour^2)
boros = c("BRONX", "BROOKLYN", "MANHATTAN", "QUEENS", "STATEN ISLAND")
preds = data.frame(matrix(ncol = 5, nrow = 24))
r2 = c(0,0,0,0,0)
names(preds) = boros
for (i in 1:5)  {
data = inci_by_hour[inci_by_hour$boro == boros[i],]
quad_mod = lm(count ~ hour +
                hour2, data=data)
preds[,i] = predict(quad_mod, inci_by_hour[inci_by_hour$boro==boros[i],])
r2[i] = summary(quad_mod)$r.squared
}

preds$hour = c(0:23)
preds = preds %>%
  pivot_longer(!hour, names_to = "boro", values_to = "pred")

# replot incident timing data with model overlay
```

```r
ggplot(inci_by_hour, aes(x=hour,y=count, color=boro)) +
  geom_point() +
  geom_line(preds, mapping = aes(x = hour, y = pred, color = boro ))
```



The number of incidents seems to vary quadratically within each borough, with different coeffecients and degree of difference for each borough.

boros	r2
BRONX	0.923
BROOKLYN	0.902
MANHATTAN	0.920
QUEENS	0.790
STATEN ISLAND	0.702

Digging Deeper

While it appears from this analysis that certain boroughs have more shooting incidents, there are also different population sizes in each borough. To see how the per capita shooting numbers looked I found historical population data for the boroughs on the NYC OpenData site.

The population data is reported for every ten years in absolute number and as a percent of the total population of NYC. For simplicity in the per capita anaylsis, I will use the average of the values from 2000, 2010, and 2020. The change in each borough is low enough that this isa reasonable approximation.

```
boro_pop = read_csv('https://data.cityofnewyork.us/resource/xywu-7bv9.csv')
```

```
## Rows: 6 Columns: 22
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): age_group, borough
## dbl (20): _1950, _1950_boro_share_of_nyc_total, _1960, _1960_boro_share_of_n...
##
## 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.
```

```
boro_pop = select(boro_pop, c('borough','_2000', '_2010', '_2020'))

boro_pop = boro_pop %>%
  rename('2000' = '_2000',
         '2010' = '_2010',
         '2020' = '_2020') %>%
  mutate(avg = rowMeans(select(boro_pop, !borough)))
```

The population data is now added to the inci_by_hour tibble so that we can create a per capita column with that data.

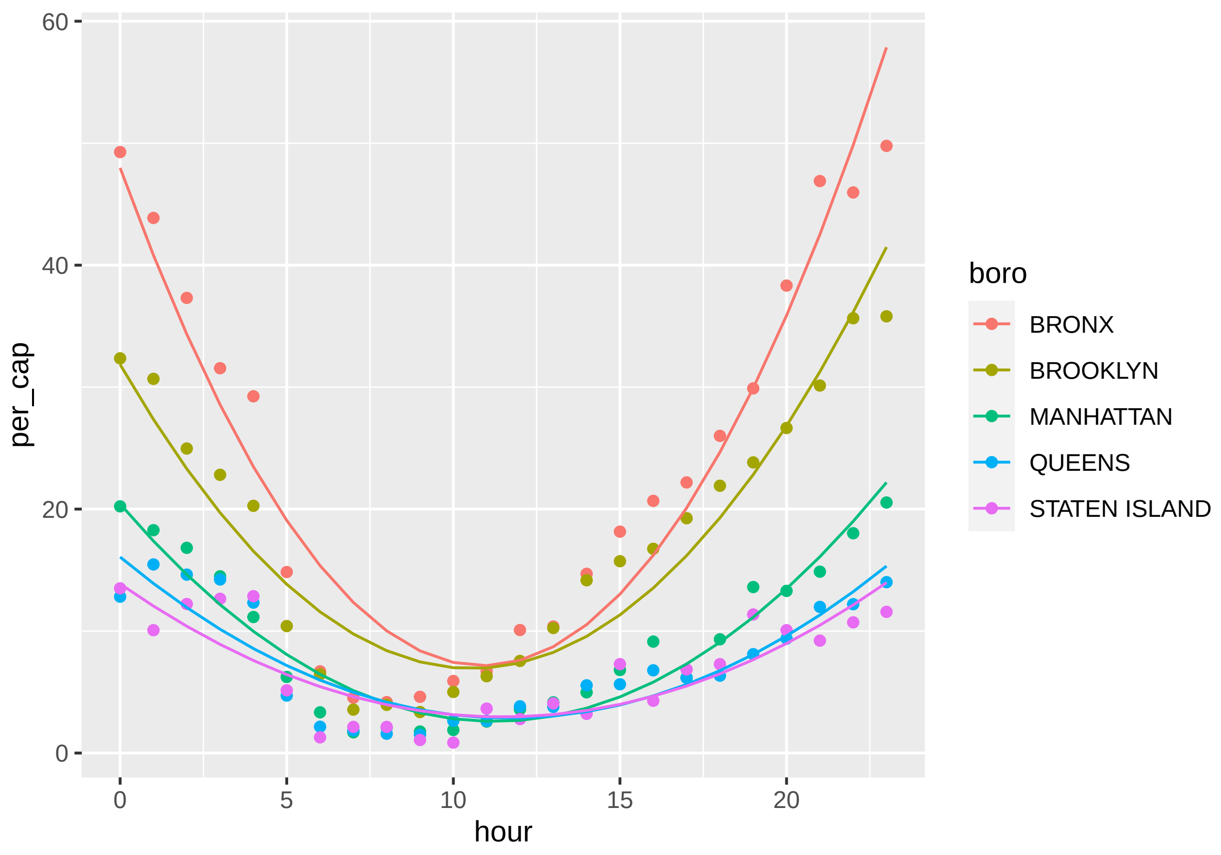
```
inci_by_hour = inci_by_hour %>%
  mutate(pop = case_when(boro == 'BRONX' ~ boro_pop$avg[2],
                         boro == 'BROOKLYN' ~ boro_pop$avg[3],
                         boro == 'MANHATTAN' ~ boro_pop$avg[4],
                         boro == 'QUEENS' ~ boro_pop$avg[5],
                         boro == 'STATEN ISLAND' ~ boro_pop$avg[6])) %>%
  mutate(per_cap = count/pop * 100000) # number of incidents per 100,000 people
```

Now we can re-do our graphing and modeling to see how much it has changed.

```
preds = data.frame(matrix(ncol = 5, nrow = 24))
r2 = c(0,0,0,0,0)
names(preds) = boros
for (i in 1:5)  {
data = inci_by_hour[inci_by_hour$boro == boros[i],]
quad_mod = lm(per_cap ~ hour +
                hour2, data=data)
preds[,i] = predict(quad_mod, inci_by_hour[inci_by_hour$boro==boros[i],])
r2[i] = summary(quad_mod)$r.squared
}

preds$hour = c(0:23)
preds = preds %>%
  pivot_longer(!hour, names_to = "boro", values_to = "pred")

# incident timing data with model overlay
ggplot(inci_by_hour, aes(x=hour,y=per_cap, color=boro)) +
  geom_point() +
  geom_line(preds, mapping = aes(x = hour, y = pred, color = boro ))
```



```
kable(data.frame(boros,r2), digits = 3)
```

boros	r2
BRONX	0.923
BROOKLYN	0.902
MANHATTAN	0.920
QUEENS	0.790
STATEN ISLAND	0.702

The per capita rates show that the apparent safety of Staten Island in the original analysis was due to the lower population of that borough. Per capita the gun violence rates in Manhattan, Queens, and Staten Island are nearly identical. The curves for the Bronx and Brooklyn switched relative positions between the raw numbers and the per capita analysis. The r squared values did not change, which is expected when transforming data by a constant scale.

Session info

```
## R version 4.3.1 (2023-06-16 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22621)
##
## 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] knitr_1.44      lubridate_1.9.3 forcats_1.0.0   stringr_1.5.0
##  [5] dplyr_1.1.3     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] bit_4.0.5         gtable_0.3.4      crayon_1.5.2      compiler_4.3.1
##  [5] tidyselect_1.2.0  parallel_4.3.1    scales_1.2.1      yaml_2.3.7
##  [9] fastmap_1.1.1     R6_2.5.1          labeling_0.4.3    generics_0.1.3
## [13] curl_5.1.0        munsell_0.5.0     pillar_1.9.0      tzdb_0.4.0
## [17] rlang_1.1.1       utf8_1.2.4        stringi_1.7.12    xfun_0.40
## [21] bit64_4.0.5       timechange_0.2.0  cli_3.6.1         withr_2.5.1
## [25] magrittr_2.0.3    digest_0.6.33     grid_4.3.1        vroom_1.6.4
## [29] rstudioapi_0.15.0 hms_1.1.3         lifecycle_1.0.3   vctrs_0.6.4
## [33] evaluate_0.22     glue_1.6.2        farver_2.1.1      fansi_1.0.5
## [37] colorspace_2.1-0  rmarkdown_2.25    tools_4.3.1       pkgconfig_2.0.3
## [41] htmltools_0.5.6.1
```