Mini Essay 2

Zhijun Zhong

Introduction

In this paper, I will be finding and discussing about the fire response time in 2018 using opendatatoronto package.

Setup

To find a data set that we are interested in.

```r
library(opendatatoronto)
library(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   3.4.4     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 becom
```

```r
library(janitor)
```

```
Attaching package: 'janitor'
```

```
The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

I am interested in a data set called “Fire incidents”. Next step is to get the package itself, and let’s look at the top 6 rows of the data set.

```
url <- "https://ckan0.cf.opendata.inter.prod-toronto.ca/dataset/64a26694-01dc-4ec3-aa87-ad

destination <- "Fire.csv"

download.file(url, destfile = destination)

fire <- read.csv(destination)
```

I am interested in the response time for different region in toronto.

cleaning the data

First, let’s clean the names for the fire data.

```
c_fire <-
  clean_names(fire)
```

Second, choose only the column called incident_station_area, tfs-alarm-time and tfs_arrival_time. Then, my data set would be much easier to look.

```
c_fire <-
  c_fire |>
  select(
    incident_station_area,
    tfs_alarm_time,
    tfs_arrival_time
  )

head(c_fire)
```

```
  incident_station_area      tfs_alarm_time    tfs_arrival_time
1                   441 2018-02-24T21:04:29 2018-02-24T21:10:11
2                   116 2018-02-24T21:24:43 2018-02-24T21:29:31
```

```
3                      221 2018-02-25T13:29:59 2018-02-25T13:36:49
4                      133 2018-02-25T14:13:39 2018-02-25T14:18:07
5                      132 2018-02-25T18:20:43 2018-02-25T18:26:19
6                      215 2018-02-25T18:31:19 2018-02-25T18:35:17
```

Now, let's see how many station areas are there.

```
c_fire$incident_station_area |>
  unique()
```

```
 [1] 441 116 221 133 132 215 235 231 332 426 225 325 226 341 421 244 141 115 415
[20] 431 331 413 314 333 311 145 143 342 443 312 223 134 214 434 423 233 114 112
[39] 224 326 212 343 135 125 315 234 324 113 142 146 313 442 222 241 345 232 121
[58] 432 425 334 411 445 243 323 435 213 422 412 123 344 111 242 321 433 245 444
[77] 211 131 322 122 335 227 346 424  NA 144
```

```
str(c_fire)
```

```
'data.frame':   29425 obs. of  3 variables:
 $ incident_station_area: num  441 116 221 133 132 215 235 231 332 426 ...
 $ tfs_alarm_time       : chr  "2018-02-24T21:04:29" "2018-02-24T21:24:43" "2018-02-25T13:29
 $ tfs_arrival_time     : chr  "2018-02-24T21:10:11" "2018-02-24T21:29:31" "2018-02-25T13:36
```

```
c_fire$tfs_alarm_time <- as.POSIXct(c_fire$tfs_alarm_time, format="%Y-%m-%dT%H:%M:%S", tz=
c_fire$tfs_arrival_time <- as.POSIXct(c_fire$tfs_arrival_time, format="%Y-%m-%dT%H:%M:%S",
```

```
c_fire$time_difference <- c_fire$tfs_arrival_time -c_fire$tfs_alarm_time

head(c_fire)
```

```
  incident_station_area      tfs_alarm_time    tfs_arrival_time time_difference
1                      441 2018-02-24 21:04:29 2018-02-24 21:10:11        342 secs
2                      116 2018-02-24 21:24:43 2018-02-24 21:29:31        288 secs
3                      221 2018-02-25 13:29:59 2018-02-25 13:36:49        410 secs
4                      133 2018-02-25 14:13:39 2018-02-25 14:18:07        268 secs
5                      132 2018-02-25 18:20:43 2018-02-25 18:26:19        336 secs
6                      215 2018-02-25 18:31:19 2018-02-25 18:35:17        238 secs
```

We have used the alarm time and arrival time. So we can drop all the columns only hold our station area and time difference.

```
c_fire <-
  c_fire |>
  select(
    incident_station_area,
    time_difference
  )

head(c_fire)
```

```
  incident_station_area time_difference
1                   441        342 secs
2                   116        288 secs
3                   221        410 secs
4                   133        268 secs
5                   132        336 secs
6                   215        238 secs
```

So, we finished cleaning the data. And We need to put our data in the output data file.

```
write_csv(
  x = c_fire, file = "cleaned_Fire.csv"
)
```

Plotting

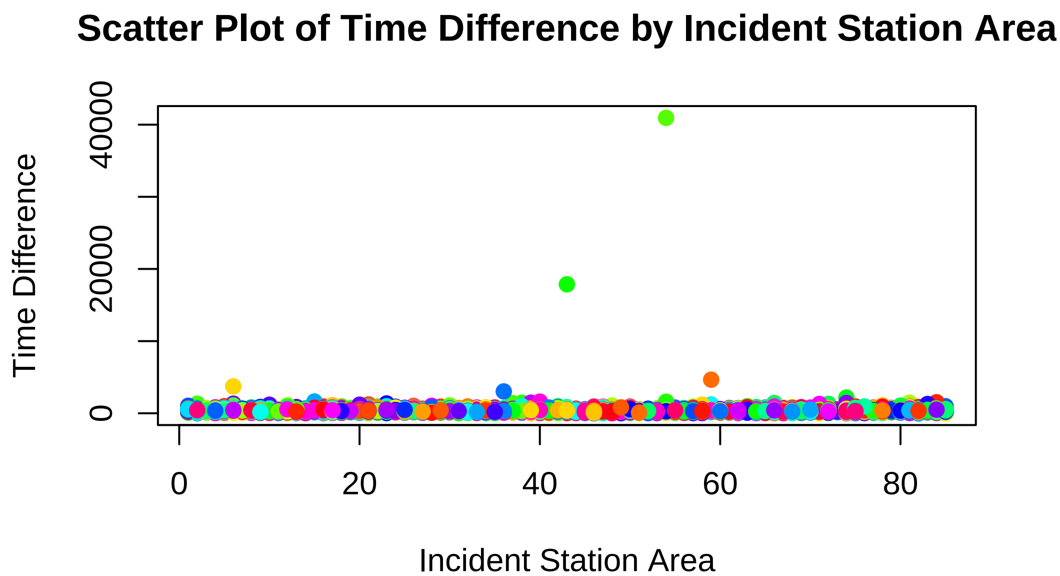
We have our data interested in the data set called c_fire. We want to use this data set to make a plot so that we could visualize the response time for each area.

```
c_fire$incident_station_area <- as.factor(c_fire$incident_station_area)
```

Since that our data set has a lot of data points, a scatter plot would be a good choice.

```
plot(c_fire$incident_station_area, c_fire$time_difference,
     xlab = "Incident Station Area",
     ylab = "Time Difference",
     main = "Scatter Plot of Time Difference by Incident Station Area",
     col = rainbow(length(unique(c_fire$incident_station_area))),
```

```
      pch = 19)
```



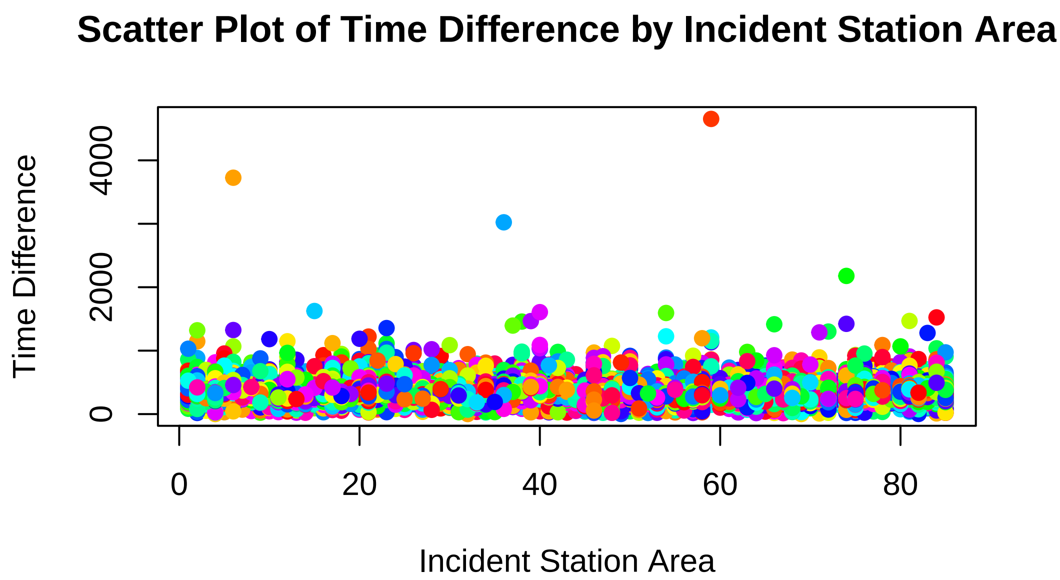
We can see that the graph is clearly destroyed by a few point which is a outlier. I want to clear those outliers by filtering the data points.

```
f_c_fire <- subset(c_fire, time_difference <= 10000)

head(f_c_fire)
```

```
  incident_station_area time_difference
1                   441        342 secs
2                   116        288 secs
3                   221        410 secs
4                   133        268 secs
5                   132        336 secs
6                   215        238 secs
```

```
f_c_fire$incident_station_area <- as.factor(f_c_fire$incident_station_area)

plot(f_c_fire$incident_station_area, f_c_fire$time_difference,
     xlab = "Incident Station Area",
     ylab = "Time Difference",
     main = "Scatter Plot of Time Difference by Incident Station Area",
```

```
    col = rainbow(length(unique(f_c_fire$incident_station_area))),
    pch = 19)
```



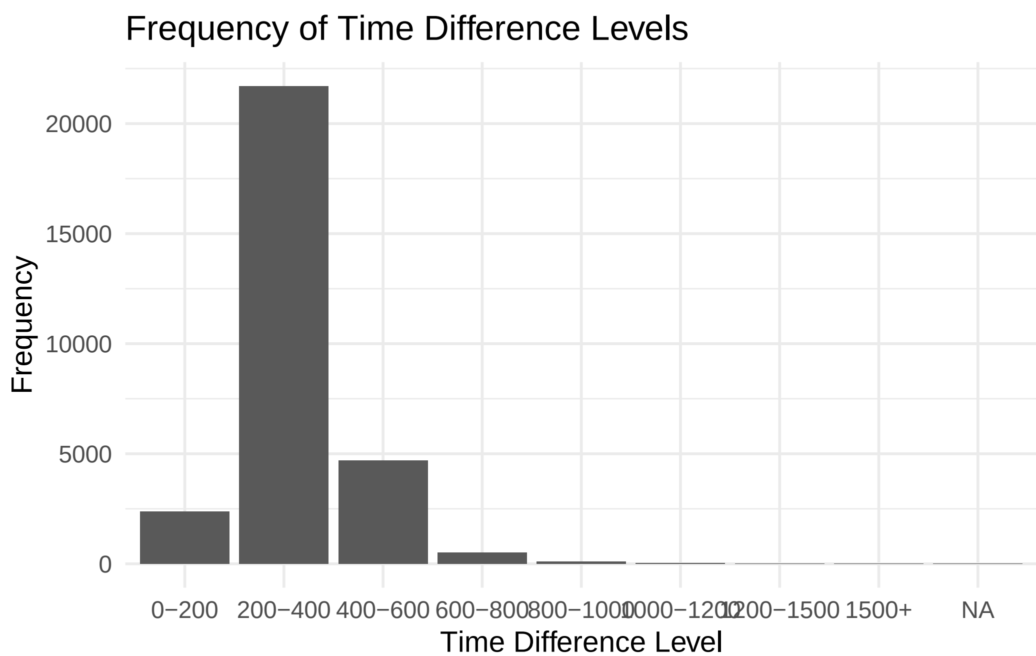
We still cannot efficiently understand the distribution of the whole thing. So I want to see how much station achieve the time difference in different level: 0 to 200 seconds, 200 to 400 seconds, 400 to 600 seconds, 600 to 800 seconds, 800 to 1000 seconds, 1000 to 1200 seconds, 1200 to 1500 seconds, 1500 seconds and more.

```
c_fire$time_difference <- as.numeric(as.character(c_fire$time_difference))

c_fire$level <- cut(c_fire$time_difference,
                    breaks = c(0, 200, 400, 600, 800, 1000, 1200, 1500, Inf),
                    labels = c("0-200", "200-400", "400-600", "600-800",
                               "800-1000", "1000-1200", "1200-1500", "1500+"),
                    right = FALSE)

head(c_fire)
```

```
  incident_station_area time_difference   level
1                   441             342 200-400
2                   116             288 200-400
3                   221             410 400-600
4                   133             268 200-400
```

```
5                    132             336  200-400
6                    215             238  200-400
```

```r
library(ggplot2)

ggplot(c_fire, aes(x = level)) +
  geom_bar() +
  theme_minimal() +
  labs(title = "Frequency of Time Difference Levels",
       x = "Time Difference Level",
       y = "Frequency") +
  scale_x_discrete(name = "Time Difference Level") +
  scale_y_continuous(name = "Frequency")
```



Analyze the data

We can see that on the graph almost all the data points lies in the level of 200 seconds to 400 seconds which is about 5 minutes. However, there is a few outliers which has its level over 2000 second shown in the table below. These maybe human errors for example, maybe someone forget to record the time of arrival. So I would ignore those data on the graph.

```r
outliers <- subset(c_fire, time_difference > 2000)

head(outliers)
```

```
      incident_station_area time_difference level
3211                    245           17871 1500+
5707                    425            2178 1500+
9227                    326           40942 1500+
26024                   233            3023 1500+
26501                   116            3726 1500+
26839                   335            4653 1500+
```

We can see that the graph shown a uniform distribution, by dividing the number of rows whose level is 200 to 400 by the total number of rows. We get 73.7%, meaning that 73.7% of all the fire response time is between 200 seconds and 400 seconds. And 81.8% of all fire response time is under 400 seconds which is 6.6 minutes.

```r
nrow(subset(c_fire, level == "200-400"))/nrow(c_fire)
```

```
[1] 0.73774
```

```r
(nrow(subset(c_fire, level == "200-400"))+nrow(subset(c_fire, level == "0-200")))/nrow(c_f
```

```
[1] 0.8182158
```

```r
(nrow(subset(c_fire, level == "200-400"))+nrow(subset(c_fire, level == "0-200"))+nrow(subs
```

```
[1] 0.977808
```

Conclusion

We can see from the graphs that we make and the data that we calculated that over 80% of time Toronto fire team in 2018 will arrive on scene within 6.67 minutes. It also shown that 98% of the time fire team will arrive on scene within 10 minutes. So if we encountered a fire, even with the unluckiest case, fire team is always able to be on cite in 10 minutes. We still have to know basic survival knowledge in order to protect ourselves till fire fighters arrive.