TriMet Analysis

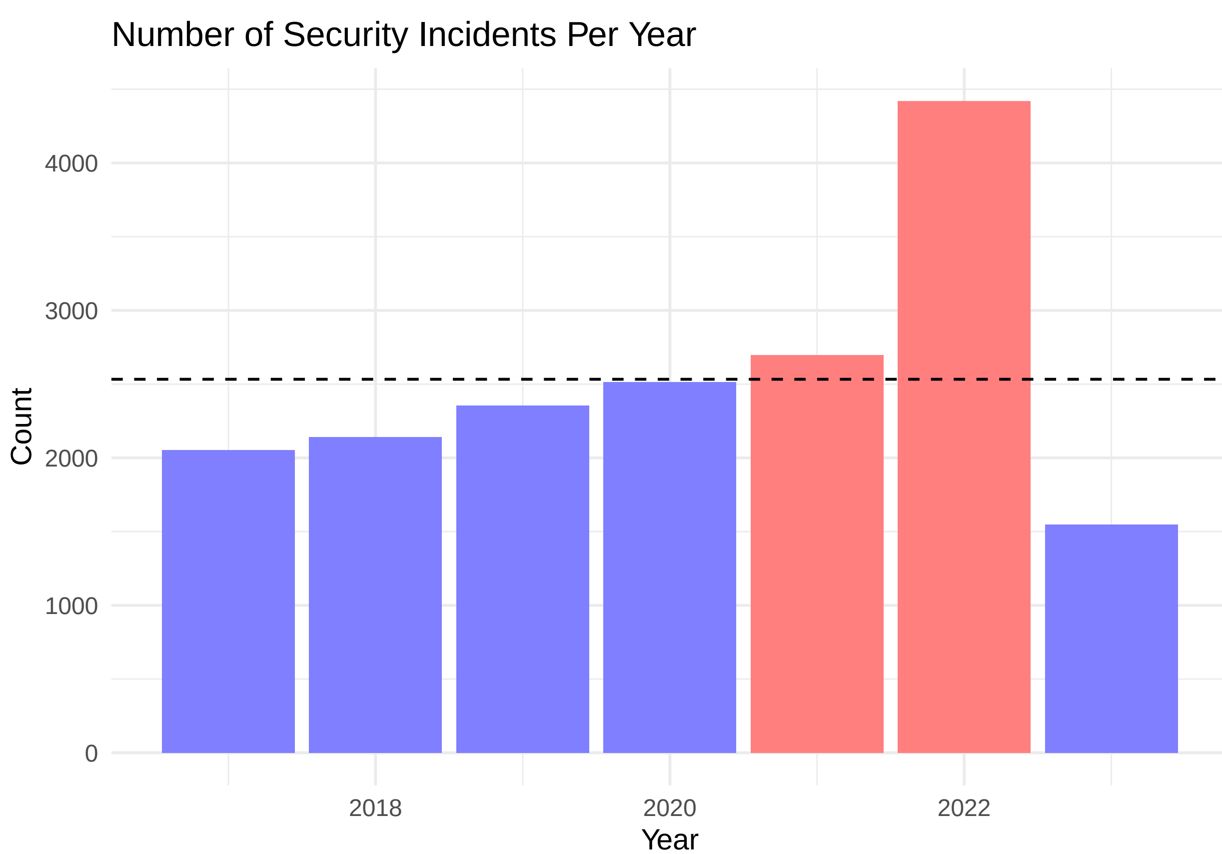
Karol Orozco, Corey Cassell, Justus Eaglesmith, Charles Hanks, & CorDarryl Hall

2023-06-06

Exploratory Data Analysis of Trimet Security Data, 2017 - 2023

Yearly Counts

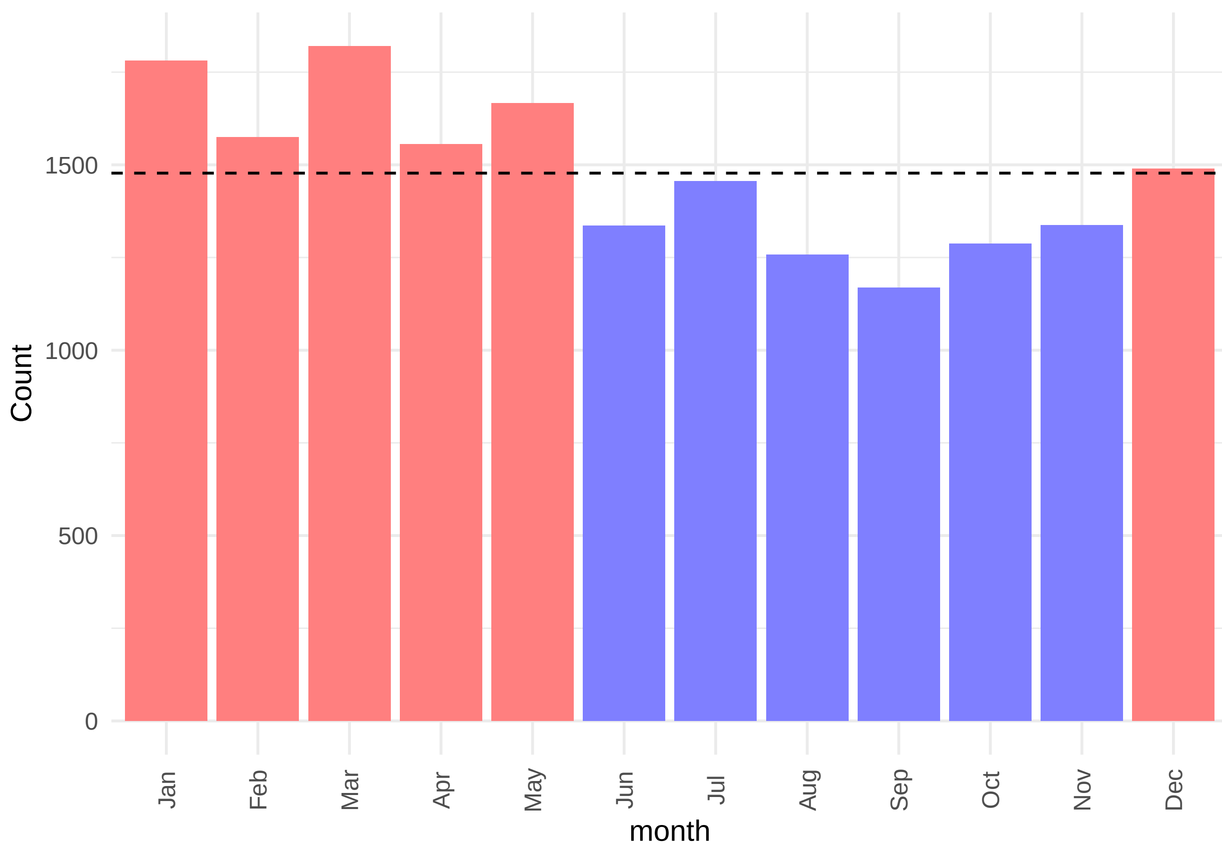
```r
# Plot yearly counts
ggplot(df_year_counts, aes(x = year, y = n, fill = color)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  geom_hline(aes(yintercept = avg), linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("Above Average" = "#FF7F7F", "Below Average" = "#7F7FFF")) +
  theme_minimal() +
  labs(x = "Year", y = "Count", fill = "", title = "Number of Security Incidents Per Year")
```



The incidents are increasing annually so it is important to get a handle on this. Especially considering 2022 had tremendously more results than 2021. A better plot for this would be using a timeseries plot.

Monthly Counts

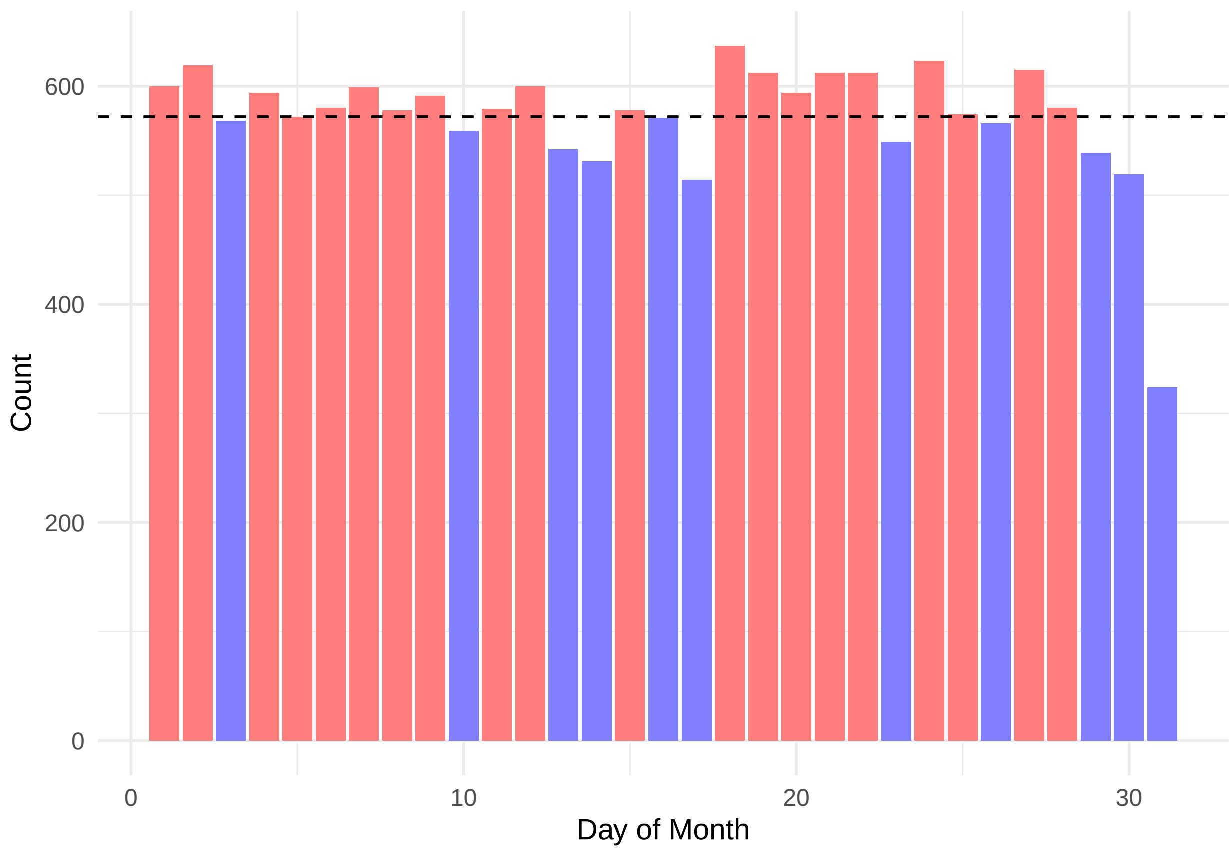
```r
# Calculate counts and average
df_month_counts <- df %>%
  count(month) %>%
  mutate(avg = mean(n),
         color = ifelse(n > avg, "Above Average", "Below Average"))

ggplot(df_month_counts, aes(x = month, y = n, fill = color)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  geom_hline(aes(yintercept = avg), linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("Above Average" = "#FF7F7F", "Below Average" = "#7F7FFF")) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
  labs(y = "Count", fill = "")
```



We should focus on the months Dec - May as those have the most incidents and are above the average. It would be valuable to understand what could be causing the increase in these specific months as well.

Daily Counts

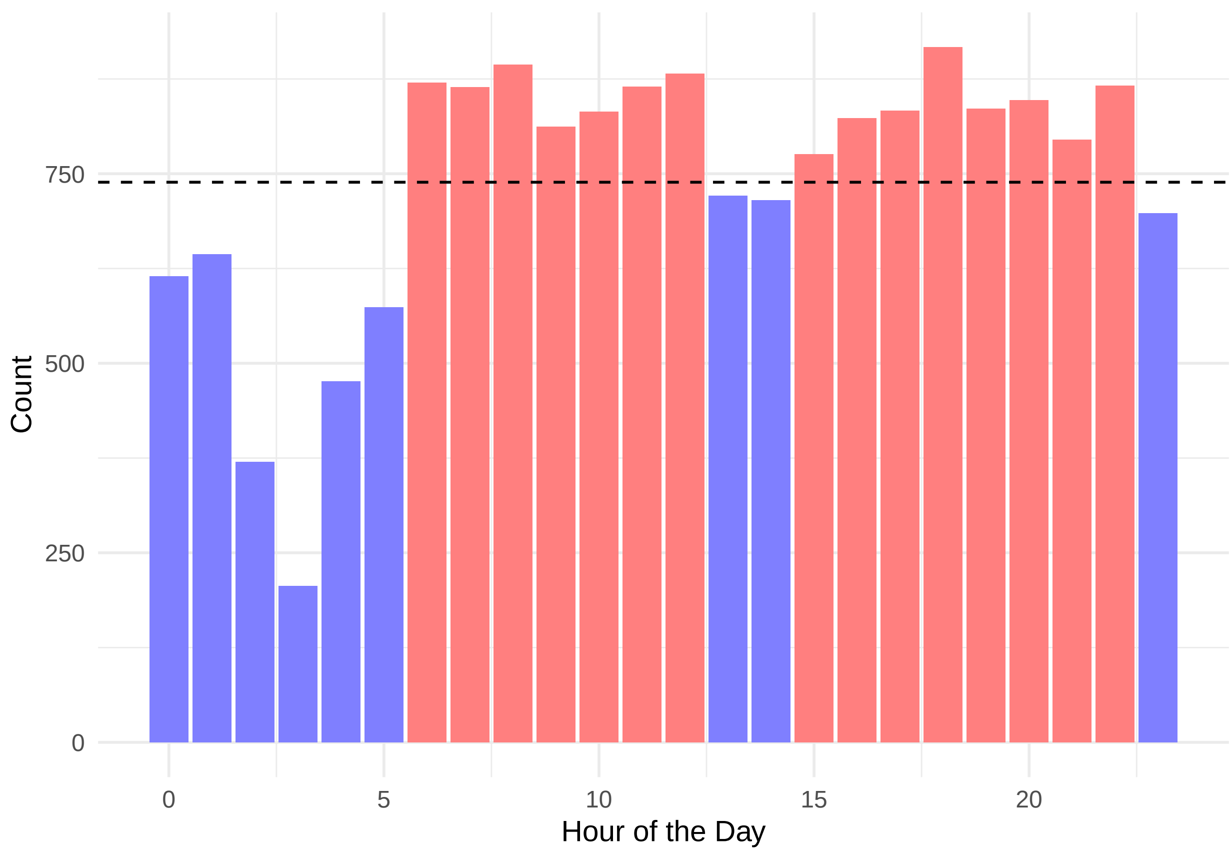
```r
# Calculate counts and average for each day
df_day_counts <- df %>%
  count(day) %>%
  mutate(avg = mean(n),
         color = ifelse(n > avg, "Above Average", "Below Average"))

# daily counts
ggplot(df_day_counts, aes(x = day, y = n, fill = color)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  geom_hline(aes(yintercept = avg), linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("Above Average" = "#FF7F7F", "Below Average" = "#7F7FFF")) +
  theme_minimal() +
  labs(x = "Day of Month", y = "Count", fill = "")
```



Days are a little sporatic but it looks like earlier in the month and later in the months there are more incidents, first and last week of the month specifically.

Hourly Incidents

```r
#Hourly incidents
library(lubridate)

# Create a new hour field
df$hour <- hour(as.POSIXct(df$incident_date, format="%m/%d/%Y %H:%M"))

# Calculate counts and average for each hour
df_hour_counts <- df %>%
  count(hour) %>%
  mutate(avg = mean(n),
         color = ifelse(n > avg, "Above Average", "Below Average"))

# Plot hourly counts
ggplot(df_hour_counts, aes(x = hour, y = n, fill = color)) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  geom_hline(aes(yintercept = avg), linetype = "dashed", color = "black") +
  scale_fill_manual(values = c("Above Average" = "#FF7F7F", "Below Average" = "#7F7FFF")) +
  theme_minimal() +
  labs(x = "Hour of the Day", y = "Count", fill = "")
```



Clearly it shows that many incidents occur at hour 6-12 and 3-12a. Makes sense as most people are commuting or using the transportation services in the morning before work/school and after work/school.

```r
#Parsing text to extract common themes amoungst reported incidents
comments_corpus <- Corpus(VectorSource(df$comments))
```

```r
comments_corpus <- tm_map(comments_corpus, content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(comments_corpus, content_transformer(tolower)):
## transformation drops documents
```

```r
comments_corpus <- tm_map(comments_corpus, removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(comments_corpus, removePunctuation):
## transformation drops documents
```

```r
comments_corpus <- tm_map(comments_corpus, removeNumbers)
```

```
## Warning in tm_map.SimpleCorpus(comments_corpus, removeNumbers): transformation
## drops documents
```

```r
comments_corpus <- tm_map(comments_corpus, removeWords, stopwords("english"))
```

```
## Warning in tm_map.SimpleCorpus(comments_corpus, removeWords,
## stopwords("english")): transformation drops documents
```

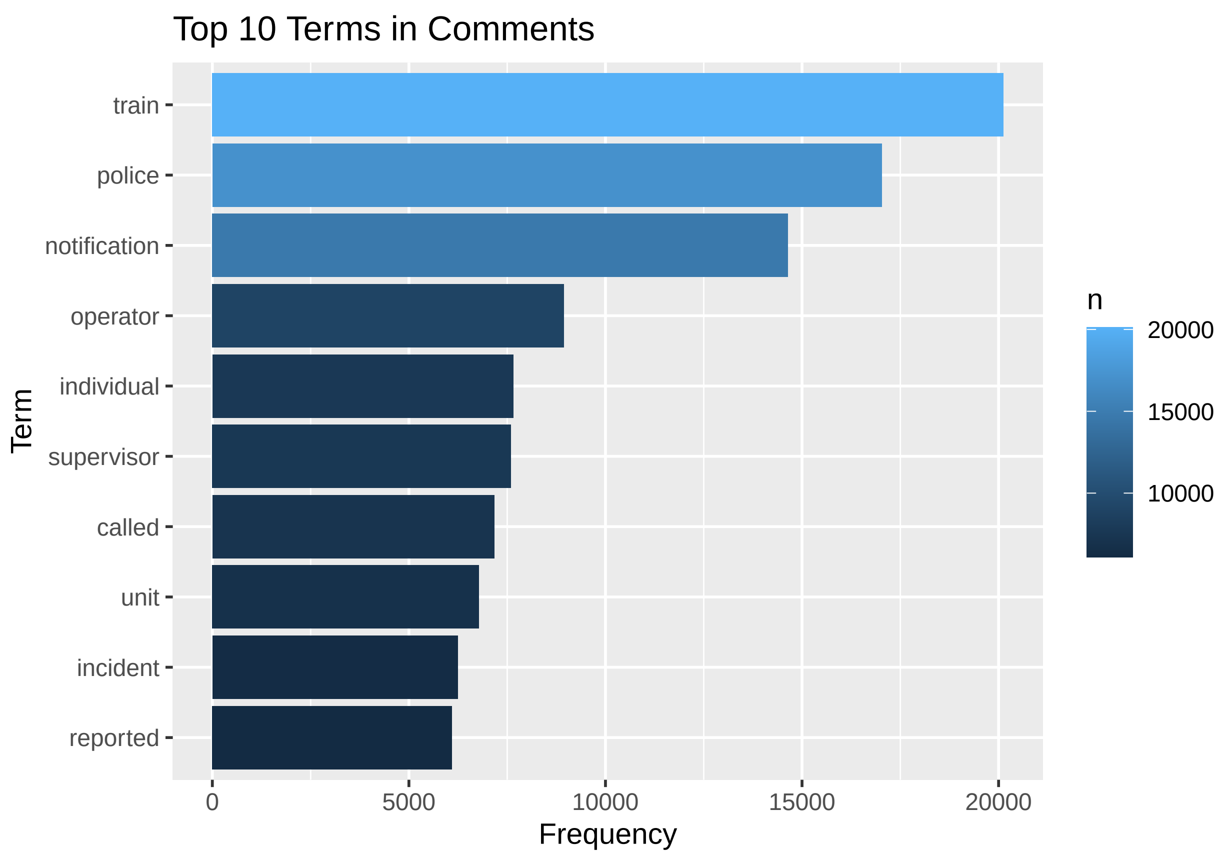
```r
comments_corpus <- tm_map(comments_corpus, stemDocument)
```

```
## Warning in tm_map.SimpleCorpus(comments_corpus, stemDocument): transformation
## drops documents
```

```r
library(tidytext)
# Converting the text to lower case
df$comments <- tolower(df$comments)

# Removing punctuation, numbers, stop words and white spaces
df$comments <- removePunctuation(df$comments)
df$comments <- removeNumbers(df$comments)
df$comments <- removeWords(df$comments, stopwords("english"))
df$comments <- stripWhitespace(df$comments)

# Tokenizing the words
df_tokens <- df %>%
  unnest_tokens(word, comments)

# Counting the frequency of each word
df_word_counts <- df_tokens %>%
  count(word, sort = TRUE)

# Filtering out words with less than 3 characters
df_word_counts <- df_word_counts[nchar(df_word_counts$word) > 2, ]

# Displaying the top 10 words
top_10_words <- df_word_counts %>%
  top_n(10) %>%
  mutate(word = reorder(word, n))
```

```
## Selecting by n
```

```
ggplot(top_10_words) +
  geom_col(aes(x = word, y = n, fill = n)) +
  labs(x = "Term", y = "Frequency", title = "Top 10 Terms in Comments") +
  coord_flip()
```



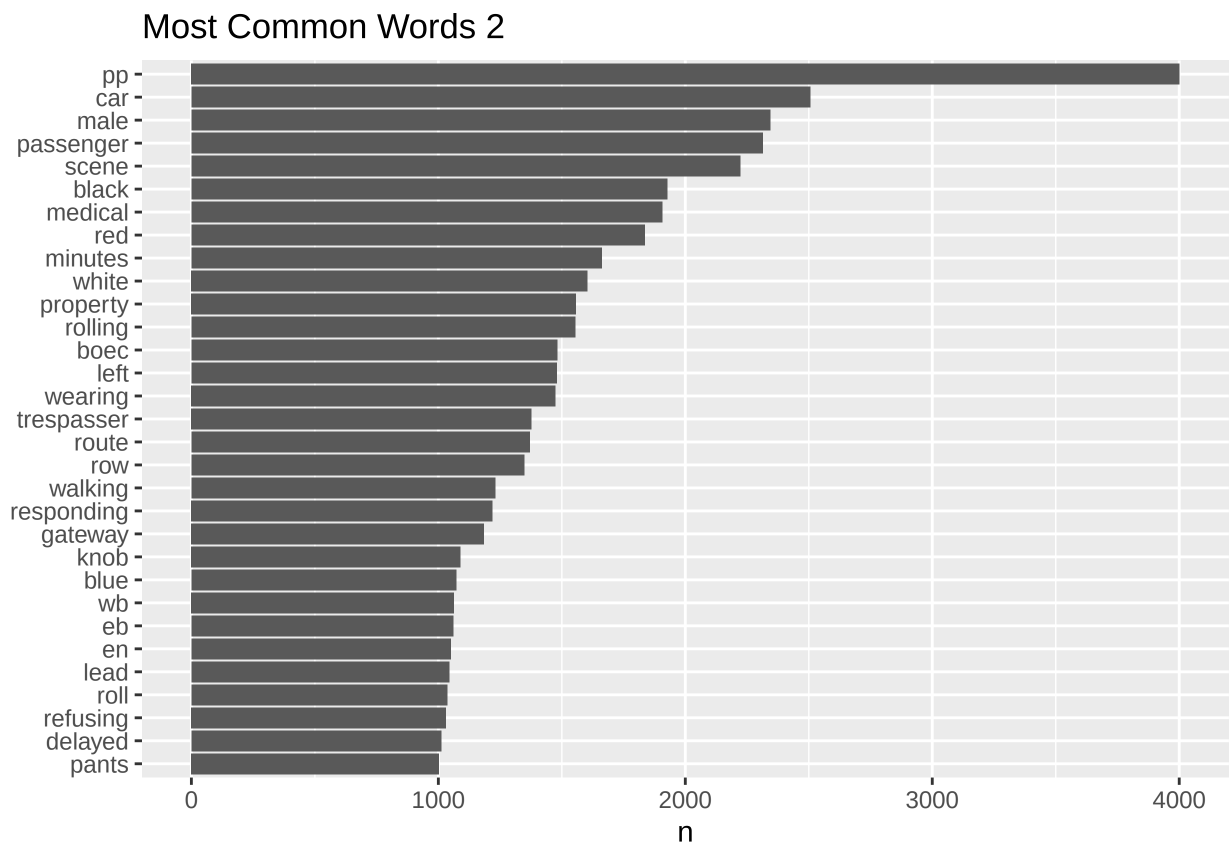
Removing common words:

```
dtm <- df %>%
unnest_tokens(word, comments) %>%
anti_join(stop_words) %>% # get rid of stop words
filter(!(word %in% c("train","notification", "police", "reports","cleared","trains","time","ave","due",
count(incident_id, word) %>%
group_by(incident_id) %>%
mutate(freq = n/sum(n)) %>%
mutate(exists = (n>0)) %>%
ungroup %>%
group_by(word) %>%
mutate(total = sum(n))
```

```
## Joining with `by = join_by(word)`
```

```
dtm %>%
count(word, sort = TRUE) %>%
```

```
  filter(n > 1000) %>%
    ggplot(aes(x = n , y= reorder(word,n))) + geom_col() + labs(y = NULL) + labs(title = "Most Common Wo
```



Incidents occur mostly on the train it appears, however we should look at the next most common phrases or nouns/adjectives to get a better understanding.

Analyzing Security Incidents at Night

What type of security incidents occurs most frequently as night ? This will require subcategory per each incident. At present in this dataset, 82% of incidents have subcategory of 'other'. We could generate subcategories through text analysis of `comments` column.

```
df %>% group_by(subtype_desc) %>% count()
```

```
## # A tibble: 20 x 2
## # Groups:   subtype_desc [20]
##    subtype_desc                   n
##    <chr>                      <int>
##  1 Assault-Employee             254
##  2 Bomb                           6
##  3 Facility                      31
##  4 Fight                        260
##  5 Hijack                         1
```

```
##   6 Homicide                           5
##   7 Hostage                            1
##   8 Park & Ride                      199
##   9 ROW Trespasser                  1159
##  10 ROW Trespasser -Non-reportable   185
##  11 Robbery                           47
##  12 Robbery w/weapon                   2
##  13 Suspicious Package                17
##  14 TVM Break-in                      29
##  15 Theft to Gain Access              12
##  16 Tow (Non-TriMet Vehicle)         312
##  17 Vandalism                        391
##  18 WES                                1
##  19 Weapon                           225
##  20 [Other]                        14594
```

```r
14591/nrow(df)
```

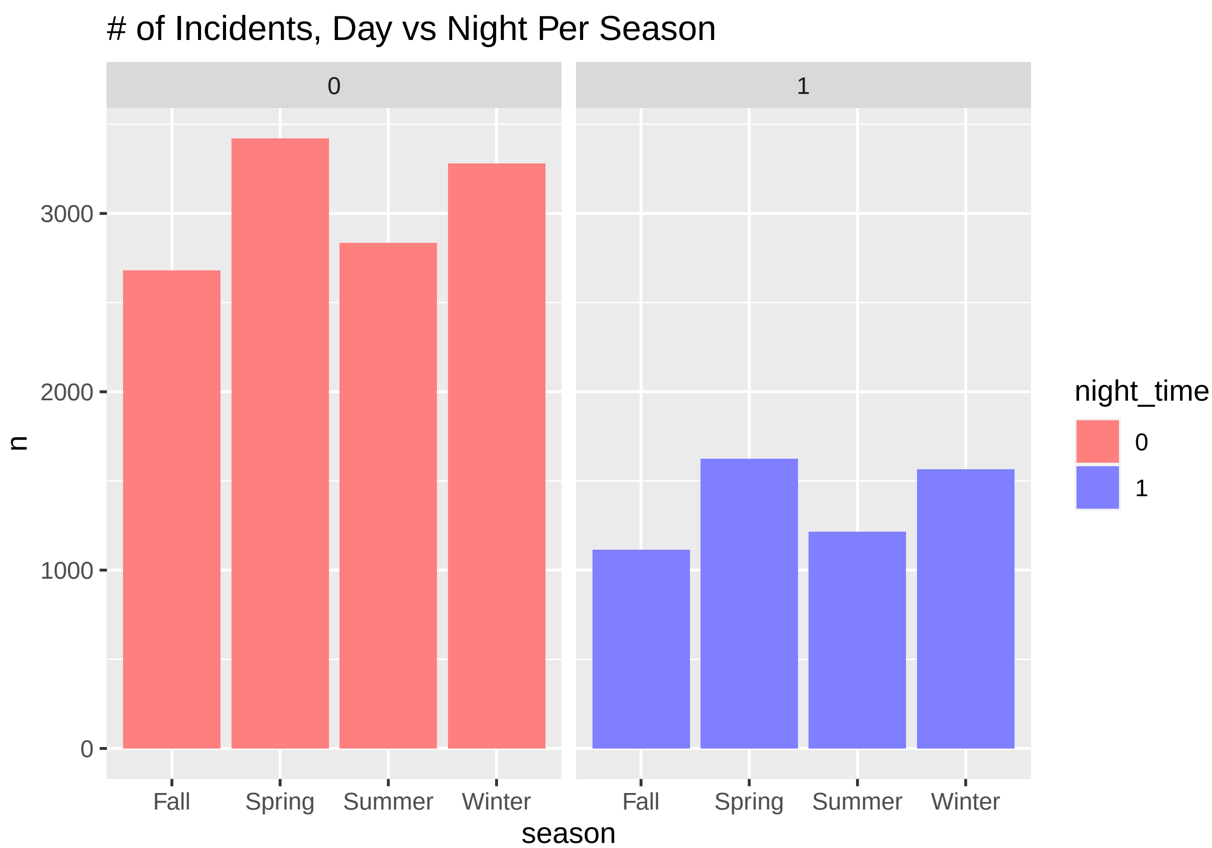
```
## [1] 0.822909
```

Winter : December - February Spring : April - June Summer : July - August Fall : September - November

```r
df = df %>% mutate(season = factor(case_when(
                          month %in% c('Dec','Jan','Feb') ~ 'Winter',
                          month %in% c('Mar','Apr', 'May') ~ 'Spring',
                          month %in% c('Jun', 'Jul', 'Aug') ~ 'Summer',
                          month %in% c('Sep', 'Oct','Nov') ~ 'Fall')),
              night_time = factor(ifelse(hour >= 20 | hour <= 4, 1, 0)))
```

Approximating nighttime as between the hours of 20:00 and 05:00, how many incidents occur at night vs during day?

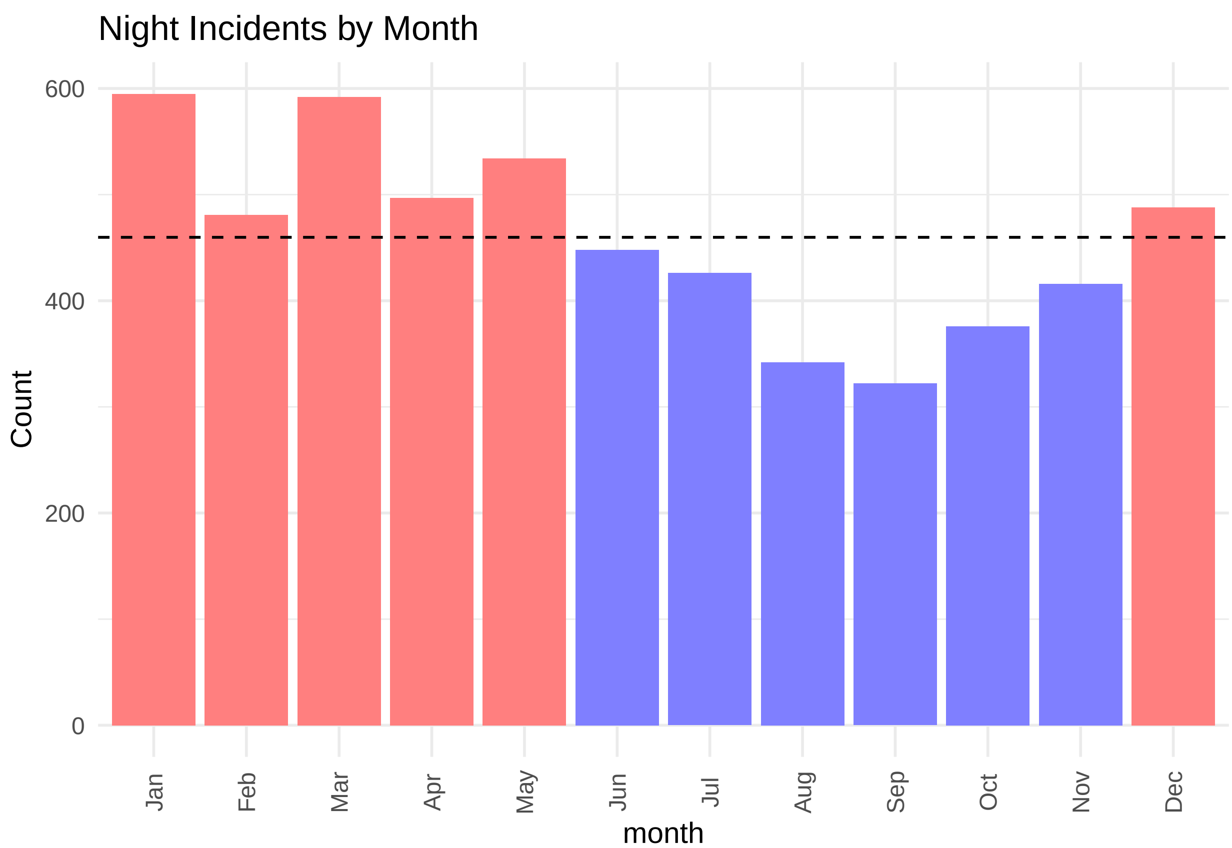
```r
df %>% group_by(night_time, season) %>% count()  %>%
            ggplot(aes(x = season, y = n, fill = night_time)) + geom_col() + facet_grid(~ night_time) +
            scale_discrete_manual(aesthetics = c("fill"), values = c("#FF7F7F", "#7F7FFF")) + labs(titl
```



We see a decrease in night time incidents during summer time. perhaps this suggests that more incidents occur when more people are using TriMet services due to weather. For example, people seeking shelter in max trains due to cold and wet conditions.

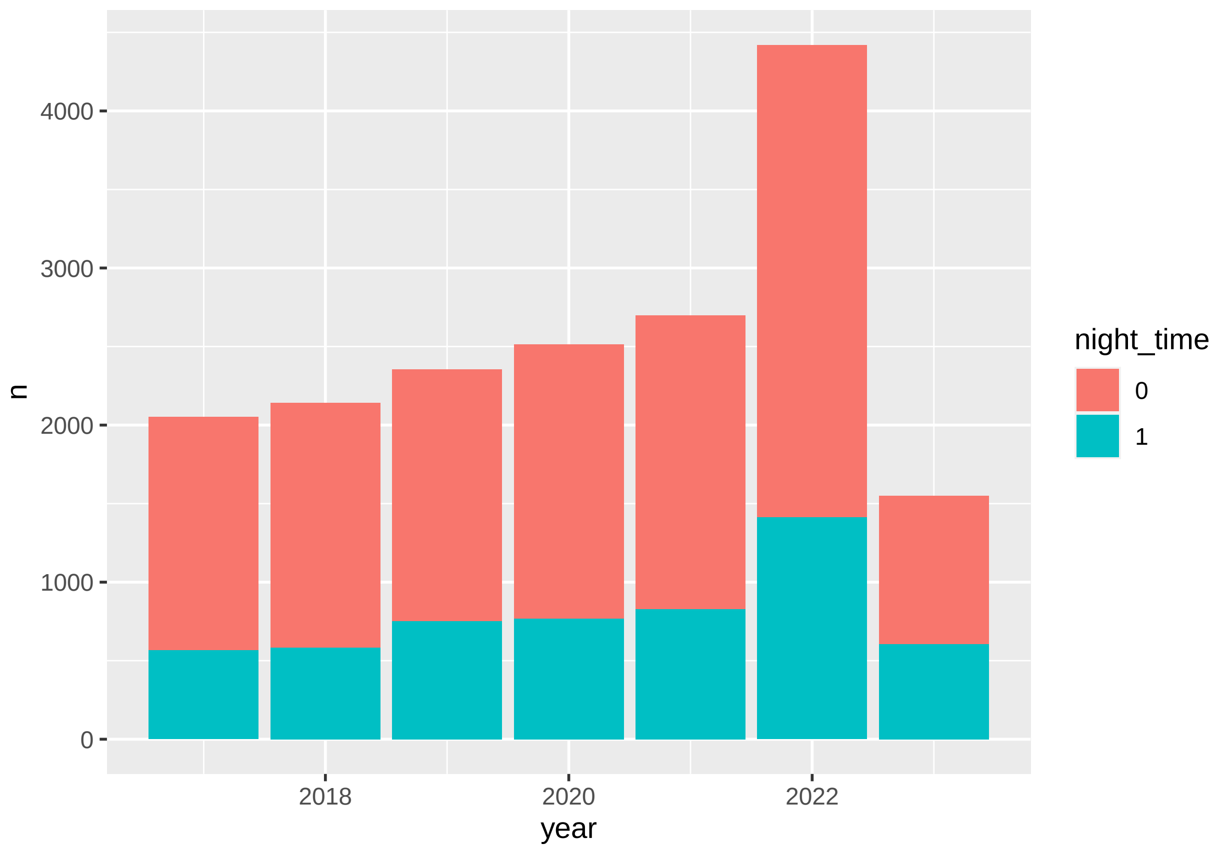
What is spread of night incidents in one year?

```
ggplot(df %>% filter(night_time == 1) %>%
  count(month) %>%
  mutate(avg = mean(n),
         color = ifelse(n > avg, "Above Average", "Below Average")), aes(x = month, y = n, fill = color)
        geom_bar(stat = "identity", show.legend = FALSE) +
        geom_hline(aes(yintercept = avg), linetype = "dashed", color = "black") +
        scale_fill_manual(values = c("Above Average" = "#FF7F7F", "Below Average" = "#7F7FFF")) +
        theme_minimal() +
        theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
        labs(y = "Count", fill = "") + labs(title = "Night Incidents by Month")
```



In summer and fall we see a decrease in incidents at night.

```
df %>% group_by(year, night_time) %>% count() %>% ggplot(aes(x = year, y = n, fill = night_time)) + geom
```



The number of incidents at night is proportional to total number of incidents per year.

Where are the most incidents occurring at night?

```
df %>% filter(night_time == 1) %>%
  group_by(location, type_desc ) %>%
    count() %>%
      arrange(desc(n))
```

```
## # A tibble: 164 x 3
## # Groups:   location, type_desc [164]
##    location           type_desc     n
##    <chr>              <chr>     <int>
##  1 Gateway Tc         Security    380
##  2 Elmonica/Sw 170th  Security    351
##  3 Cleveland Avenue   Security    294
##  4 Ruby Jct/197th Ave Security    282
##  5 Rose Quarter Tc    Security    211
##  6 Willow Crk/185th Tc Security   154
##  7 Hollywood/42nd Ave Security    139
##  8 <NA>               Security    136
##  9 82nd Avenue        Security    116
## 10 Beaverton Tc       Security    105
## # i 154 more rows
```

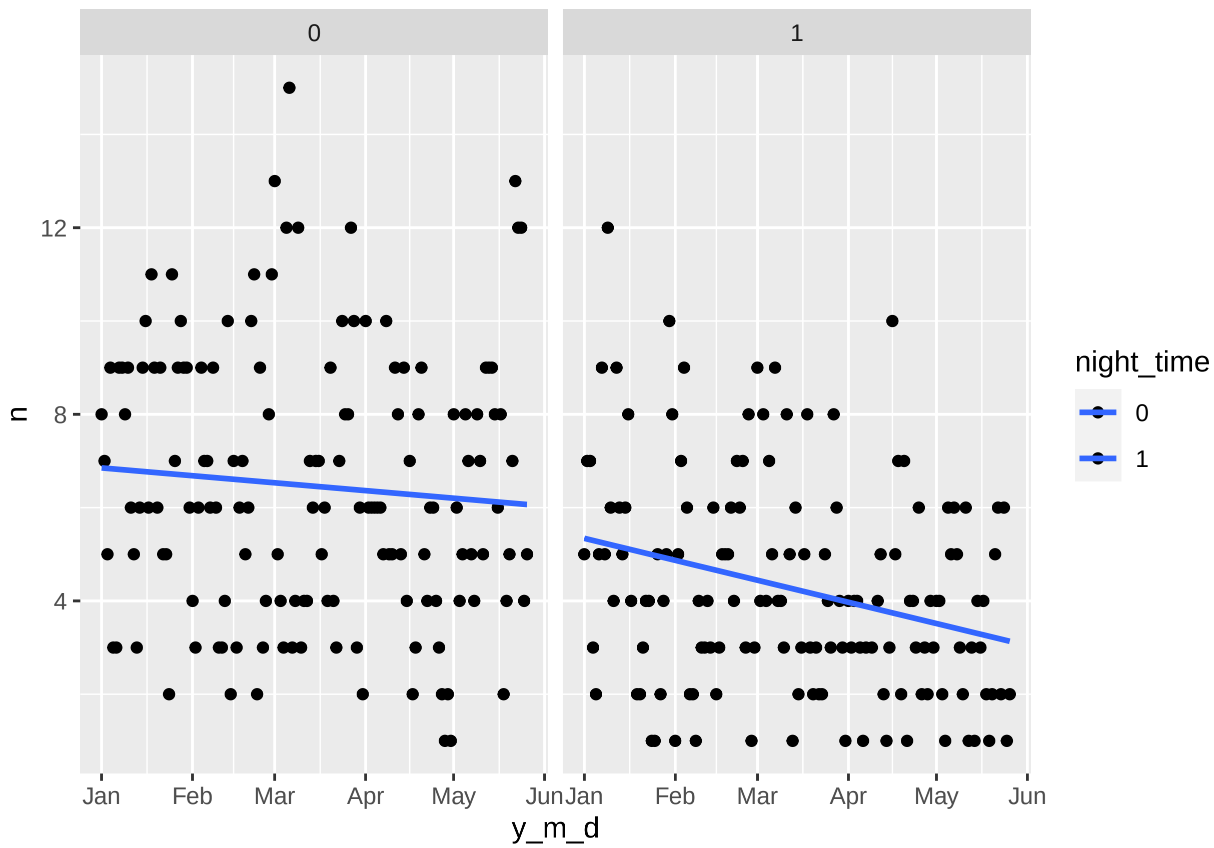
Elmonica is where the MAX trains are stored and serviced - most trains in the morning start here. It would make sense that security personnel are reporting from here. Gateway is a hot spot for all types of activity.

`Cleveland Avenue` we assume is in reference to MAX station in Gresham (final stop for the Blue Line) Cleveland is mentioned as a place where 'sleepers' are found.

In 2023, have we seen a downward trend in incidents at night given the increased presence of security personnel starting in March 2023?

```
df = df %>% mutate(y_m_d = date(as.POSIXct(incident_date, format="%m/%d/%Y %H:%M"))) # adding date only

df %>% filter(year == 2023) %>% group_by(y_m_d, night_time) %>% count() %>%  ggplot(aes(x = y_m_d, y = 
  geom_smooth(method = 'lm', se = FALSE) +
  facet_grid(~night_time)
```

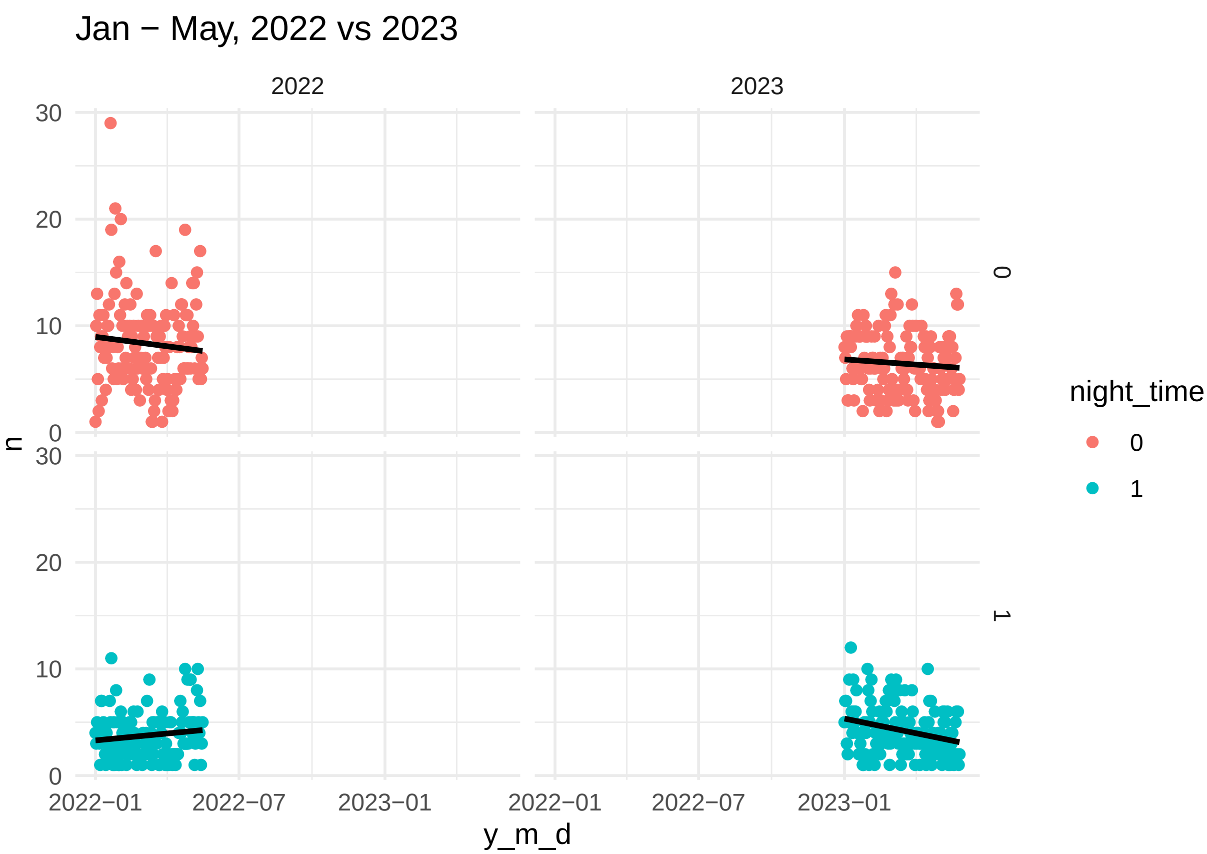
```
## `geom_smooth()` using formula = 'y ~ x'
```



What if we compare the same time interval, one year ago?

```
df %>% filter(year %in% c(2022,2023))%>%
  filter((y_m_d >= "2022-01-01" & y_m_d <= "2022-05-16") | y_m_d >= "2023-01-01")%>%
  group_by(y_m_d,year, night_time) %>% count() %>%  ggplot(aes(x = y_m_d, y = n, color = night_time))  
  geom_smooth(method = 'lm', se = FALSE, color = "black") +
  facet_grid(night_time~year) +
  theme_minimal() + labs(title = "Jan - May, 2022 vs 2023")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Nighttime incidents were trending upward in Jan - May 2022, and now they are trending downward Jan - May 2023.

A Look at location data

I want to filter all the incidents that occurred on the max blue line (not necessarily on the blue train itself, but on and around the stops it goes on) I need to match the blue line stops to the locations provided in the data:

```
unique(df$location) %>% sort()
```

```
##   [1] "102nd Avenue"            "11th"
##   [3] "11th Avenue"             "122nd Avenue"
##   [5] "13th"                    "148th Avenue"
##   [7] "162nd Avenue"            "172nd Avenue"
##   [9] "181st Avenue"            "185th"
##  [11] "1st Ave"                 "205th"
##  [13] "3rd Avenue"              "5th"
##  [15] "60th"                    "60th Avenue"
##  [17] "6th"                     "7th"
##  [19] "7th Avenue"              "82nd"
##  [21] "82nd Avenue"             "96th"
##  [23] "Albina/Mississippi"      "Beaverton Central"
##  [25] "Beaverton Creek"         "Beaverton Tc"
##  [27] "Broadway"                "Broadway Siding"
```

```
##  [29] "Burnside"                    "Cascades"
##  [31] "Cascades Pkwy"               "Cesar Chavez Blvd"
##  [33] "City Hall/5th & Jefferson" "Civic Drive"
##  [35] "Clackamas Town Center Tc"  "Cleveland Avenue"
##  [37] "Clinton Pocket Track"      "Clinton St/Se 12th Ave"
##  [39] "Convention Center"         "Couch"
##  [41] "Davis"                     "Delta Park/Vanport"
##  [43] "Denver"                    "East Portal"
##  [45] "Eastside Max"              "Elam Young Pkwy"
##  [47] "Elmonica Yard"             "Elmonica Yl3"
##  [49] "Elmonica/Sw 170th"         "Expo Center"
##  [51] "Fairplex/Airport"          "Flavel"
##  [53] "Galleria/10th Ave"         "Gateway Sig 74 Loop"
##  [55] "Gateway Tc"                "Glisan"
##  [57] "Goose Hollow/Sw Jefferson" "Grand"
##  [59] "Gresham Central Tc"        "Gresham City Hall"
##  [61] "Hall/Nimbus"               "Harbor"
##  [63] "Hatfield Govt Center"      "Hawthorn Farm"
##  [65] "Hillsboro Central Tc"      "Holladay"
##  [67] "Hollywood Tc Pocket Track" "Hollywood/42nd Ave"
##  [69] "Hwy-217"                   "I-205/Burnside"
##  [71] "I-205/Monterey"            "Interstate"
##  [73] "Interstate/Rose Quarter"   "Jackson St Turnaround"
##  [75] "Jackson St/6th Ave"        "Kenton/N Denver Ave"
##  [77] "Killingsworth St"          "Kings Hill/Salmon"
##  [79] "Knight"                    "Lents/Se Foster Rd"
##  [81] "Library/9th Ave"           "Lincoln St/Sw 3rd Ave"
##  [83] "Lloyd Center/11th"         "Lloyd Center/Dbltree"
##  [85] "Lombard Tc"                "Main"
##  [87] "Main St Pocket Track"      "Mall/Sw 4th Ave"
##  [89] "Mall/Sw 5th Ave"           "Market"
##  [91] "Mcloughlin"                "Merlo Rd/158th"
##  [93] "Mill"                      "Millikan Way"
##  [95] "Milwaukie/Main St"         "Moody"
##  [97] "Morrison"                  "Morrison Sw"
##  [99] "Mt Hood Ave"               "North Terminal E Pkt Trk"
## [101] "North Terminal W Pkt Trk"  "Nw 5th & Couch St"
## [103] "Nw 6th & Davis St"         "Oak Street"
## [105] "Old Town/ Chinatown"       "Omsi/Se Water"
## [107] "Oregon"                    "Orenco"
## [109] "Overlook Park"             "Parkrose/ Sumner Stn"
## [111] "Pioneer Courthouse/6th"    "Pioneer Place/Sw 5th Ave"
## [113] "Pioneer Square North"      "Pioneer Square South"
## [115] "Portland Airport"          "Powell"
## [117] "Prescott St"               "Providence Park"
## [119] "Psu"                       "Psu South/5th & Jackson"
## [121] "Psu South/6th & College"   "Psu Urban Ctr/5th & Mill"
## [123] "Psu Urban Ctr/6&Montg"     "Quatama"
## [125] "Quatama Park & Ride"       "Rockwood/188th Ave"
## [127] "Rocky Butte"               "Rosa Parks"
## [129] "Rose Quarter Tc"           "Ruby Jct/197th Ave"
## [131] "Ruby Yard"                 "Ruby Yl2"
## [133] "Ruby Yl3"                  "Se 17th Ave & Holgate Blv"
## [135] "Se 17th Ave & Rhine St"    "Se Bybee Blvd"
```

```
## [137] "Se Division St"            "Se Flavel St"
## [139] "Se Fuller Rd"               "Se Holgate Blvd"
## [141] "Se Main St"                 "Se Park Ave"
## [143] "Se Powell Blvd"             "Se Tacoma/Johnson Crk"
## [145] "Skidmore Fountain"          "South Waterfront/S Moody"
## [147] "Steel Bridge"               "Stop 3 800'"
## [149] "Sunset Tc"                  "Sw 5th & Oak St"
## [151] "Sw 6th & Madison St"        "Sw 6th & Pine St"
## [153] "Tigard Tc"                  "Tuality Hospital/Se 8th"
## [155] "Twc 76"                     "Twc 78"
## [157] "Twc 80"                     "Twc_1084a"
## [159] "Twc_120"                    "Twc_122nd Ave E/B"
## [161] "Twc_122nd Ave W/B"          "Twc_148th Ave E/B"
## [163] "Twc_148th Ave W/B"          "Twc_154"
## [165] "Twc_162nd Ave E/B"          "Twc_162nd Ave W/B"
## [167] "Twc_172nd Ave E/B"          "Twc_172nd Ave W/B"
## [169] "Twc_1760"                   "Twc_181st Ave E/B"
## [171] "Twc_181st Ave W/B"          "Twc_186a"
## [173] "Twc_18f"                    "Twc_22"
## [175] "Twc_356"                    "Twc_38"
## [177] "Twc_3rd Ave & Morisson St"  "Twc_40"
## [179] "Twc_4th Ave. & Yamhill St"  "Twc_5th Ave & Morrison St"
## [181] "Twc_6th Ave & Yamhill St."  "Twc_864a"
## [183] "Twc_924a"                   "Twc_9th Avenue E/B"
## [185] "Twc_A340"                   "Twc_A346"
## [187] "Twc_A550"                   "Twc_A552"
## [189] "Twc_A760"                   "Twc_A780"
## [191] "Twc_Broadway & Morrison"    "Twc_Galleria W/B"
## [193] "Twc_Gateway W/B"            "Twc_Goose Hollow E/B"
## [195] "Twc_Goose Hollow W/B"       "Twc_Hawthorne Farm W/B"
## [197] "Twc_Oak St. & 1st Ave."     "Twc_Quatama E/B"
## [199] "Twc_Sunset Tc E/B"          "Twc_Sunset Tc W/B"
## [201] "Twc_Yamhill St. & 1st Ave"  "Union Stn/5th & Glisan"
## [203] "Union Stn/6th & Hoyt"       "Vt Barn"
## [205] "Washington"                 "Washington Park"
## [207] "Washington/Se 12th"         "Water"
## [209] "West Portal"                "Willow Creek Tc Temp"
## [211] "Willow Crk/185th Tc"        "Yamhill"
## [213] "Yamhill District"
```

```
df %>% filter(location == "Cleveland")
```

```
## # A tibble: 0 x 21
## # i 21 variables: incident_id <dbl>, type_code <chr>, division <chr>,
## #   type_desc <chr>, subtype_code <chr>, subtype_desc <chr>,
## #   incident_date <chr>, location <chr>, intersection <chr>, direction <chr>,
## #   lift_location <lgl>, comments <chr>, subtype_t <chr>, count_t <chr>,
## #   year <dbl>, month <ord>, day <int>, hour <int>, season <fct>,
## #   night_time <fct>, y_m_d <date>
```

```
max_blue_stops = c("Hatfield Government Center",
                     "Hillsboro Central/SE 3rd Avenue",
           "Tuality Hospital/SE 8th Avenue",
```

```r
            "Washington/SE 12th Avenue",
            "Fair Complex/Hillsboro Airport",
            "Hawthorn Farm",
            "Orenco",
            "Quatama/NW 205th Avenue",
            "Willow Creek/SW 185th Avenue Transit Center",
            "Elmonica/SW 170th Avenue",
            "Merlo Road/SW 158th Avenue",
            "Beaverton Creek",
            "Millikan Way",
            "Beaverton Central",
            "Beaverton Transit Center",
            "Sunset Transit Center",
            "Washington Park",
            "Goose Hollow",
            "Providence Park",
            "Galleria SW 10th",
            "Pioneer Courthouse Square",
            "Mall/SW 4th Avenue",
            "Yamhill District",
            "Oak St",
            "Skidmore Fountain",
            "Old Town/Chinatown",
            "Rose Quarter Transit Center/Northeast 7th Avenue",
            "Convention Center",
            "NE 7th",
            "Lloyd Center/Northeast 11th Avenue",
            "Hollywood/NE 42nd Avenue Transit Center",
            "Hollywood/Northeast 42nd Avenue",
            "60th",
            "82nd",
            "Gateway/NE 99th Avenue Transit Center",
            "102nd",
            "122nd",
            "148th",
            "172nd",
            "181st",
            "Rockwood/188th Ave",
            "Ruby Junction/East 197th Avenue",
            "Civic Drive",
            "Gresham City Hall",
            "Cascade Station",
            "Mt. Hood Avenue",
            "East 181st Avenue",
            "Cleveland Avenue",
            "Gresham Central Tc",
            "Cleveland"
            )

trimet_locations = unique(df$location)

library(stringdist)
```

```
##
## Attaching package: 'stringdist'

## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
#this function to be used to adjust the names of the max blue line stop above and to match them as clos
adjust_vector <- function(vector1, vector2) {
  adjusted_vector <- sapply(vector1, function(stop) {
    closest_match <- min(stringdist(stop, vector2))
    closest_stop <- vector2[which(stringdist(stop, vector2) == closest_match)][1]
    closest_stop
  })

  adjusted_vector
}

adjusted_stops = adjust_vector(max_blue_stops, trimet_locations)


adjusted_stops = names(adjusted_stops)

hollywood_check = function(x)
{
  if (grepl("Hollywood", x))
  {
    x <- gsub("Hollywood.*", "Hollywood Tc", x)
  }
  return(x)
}

adjusted_stops = lapply(adjusted_stops, hollywood_check)

adjusted_stops = unique(adjusted_stops) #removing 1 'Hollywood Tc'

adjusted_stops[29] = "7th Avenue"

#got to change all the 60th, 12nd to include word "Avenue"
#regex pattern for 102nd, 60th:
pattern = "^\\d{2,3}[A-Za-z]{2}$"

#adding "Avenue" to each of these stops
adjusted_stops[grepl(pattern,adjusted_stops)] = paste(adjusted_stops[grepl(pattern,adjusted_stops)], "A

#removing repeated stops
repeats = c("East 181st Avenue","Cleveland")
adjusted_stops = adjusted_stops[!(adjusted_stops %in% repeats)]

#adding library stop
adjusted_stops = append(adjusted_stops, "Library/9th Ave")

adjusted_stops[21] = "Pioneer"
```

filter df with max stops:

```
# max = df %>% filter(location %in% adjusted_stops)
#
# unique(max$location) #did not successfully match with all max stops
#
# #running adjust_vector() again:
# adjusted_stops = unlist(adjusted_stops)
# adjusted_stops = names(adjust_vector(adjusted_stops, trimet_locations))
#
# Hollywood = df %>% filter(str_detect(location, "Hollywood")) #df contains 2 locations with hollywood:
#
# #did hollywood pass the filter?
# max_stops %>% filter(str_detect(location, "Hollywood")) # only 2 hollywood Tc
#
# df %>% filter(str_detect(location, "Hollywood/42nd Ave"))
#
# adjusted_stops = append(adjusted_stops, "Hollywood/42nd Ave")
#
# max_stops = df[grep(paste(adjusted_stops, collapse = "|"), df$location, ignore.case = TRUE), ]
#
# #how many locations in max_stops
# length(unique(max_stops$location)) # 36 not bad...but where did gateway tc go ?
```

This does not seem like an effective way to track the blue line, given that the events at this stops could be from a variety of different bus / train lines. What instead I look into events taht occur at transit centers?