# PLAN372: Blocked Train Crossings & The Impact on Pedestrians

Anna Fetter

2025-04-15

## Research Question:

**How do neighborhood walkability and demographic characteristics relate to the frequency and severity of blocked train crossing complaints in U.S. cities, specifically complaints that involve pedestrians climbing over or under train cars?**

### Load Packages

```
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   3.5.1     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 become errors
```

```
library(lubridate)
library(ggplot2)
library(readxl)
library(ggrepel)
```

```
## Warning: package 'ggrepel' was built under R version 4.3.3
```

```
library(purrr)
library(tidycensus)
```

```
## Warning: package 'tidycensus' was built under R version 4.3.3
```

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 4.3.3
```

```
## Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE
```

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 4.3.3
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
##   Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service>
##   OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

```
library(osmdata)
```

```
## Data (c) OpenStreetMap contributors, ODbL 1.0. https://www.openstreetmap.org/copyright
```

### Load Data

```
X22_blocked_crossings <- read_excel("data/2022_blocked_crossings.xlsx")
X23_blocked_crossings <- read_excel("data/2023_blocked_crossings.xlsx")
X24_blocked_crossings <- read_excel("data/2024_blocked_crossings.xlsx")

#join all 3 sets together, downloading more than a year at a time from the website was really finnicky

blocked_crossings <- bind_rows(X22_blocked_crossings, X23_blocked_crossings, X24_blocked_crossings)
```

## Preliminary Data Interview

### Taking the data "out for coffee"

```
summary(blocked_crossings)
```

```
##  Crossing ID            City              State              Street
##  Length:76742       Length:76742       Length:76742       Length:76742
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     County            Railroad           Date/Time
##  Length:76742       Length:76742       Min.   :2022-01-01 00:00:00.00
##  Class :character   Class :character   1st Qu.:2022-08-09 12:03:00.00
##  Mode  :character   Mode  :character   Median :2023-05-19 11:25:00.00
##                                        Mean   :2023-06-09 00:20:36.88
##                                        3rd Qu.:2024-04-09 16:49:45.00
##                                        Max.   :2024-12-30 20:24:00.00
##    Duration            Reason          Immediate Impacts  Additional Comments
```

```
##  Length:76742       Length:76742       Length:76742       Length:76742
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
#unique(blocked_crossings$Reason)
#unique(blocked_crossings$`Immediate Impacts`)
#unique(blocked_crossings$Railroad)
#unique(blocked_crossings$Duration)
```

From initial data interview, it looks like there are problems with 2-6 hour delays being listed as “2-6 hours”, “2-6 hours”’, and “2-6 hours”“. I will change them all to”2-6 hours” for sake of clean and consistent data.

```
blocked_crossings <- blocked_crossings %>%
  mutate(Duration = str_replace_all(Duration, "2-6 hours'", "2-6 hours")) %>%
  mutate(Duration = str_replace_all(Duration, "2-6 hours\"", "2-6 hours"))

unique(blocked_crossings$Duration)
```

```
## [1] "0-15 minutes"      "16-30 minutes"     "2-6 hours"
## [4] "1-2 hours"         "31-60 minutes"     "6-12 hours"
## [7] "More than one day" "12-24 hours"
```

Let’s see how many states we have listed.

```
blocked_crossings %>%
  group_by(State) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))
```

```
## # A tibble: 49 x 2
##    State Count
##    <chr> <int>
##  1 TX    19100
##  2 IL     6534
##  3 OH     6103
##  4 IN     4204
##  5 TN     3401
##  6 IA     2926
##  7 MI     2840
##  8 GA     2780
##  9 KS     2609
## 10 AL     2582
## # i 39 more rows
```

Huh, only 49 states, let’s match against an existing list of states and see who isn’t on the list

```r
# built-in state abbreviations
state_abbreviations <- state.abb

# get distinct state abbreviations
observed_states <- blocked_crossings %>%
  distinct(State) %>%
  pull(State)

# identify unknown state abbreviations
invalid_states <- setdiff(observed_states, state_abbreviations)

# view them
print(invalid_states)
```

```
## [1] "DC"
```

```r
# let's see what states are missing that are in abbreviations but not observed states
missing_states <- setdiff(state_abbreviations, observed_states)

# view missing states
print(missing_states)
```

```
## [1] "HI" "RI"
```

Based on this analysis DC is in the dataset, but Hawaii and Rhode Island are missing. A quick google search says that Hawaii has no active commercial railroads now. Rhode Island has some, but it's a super small state so not having complaints isn't a worry.

To make things easier later on, let's separate the year, month, and time into separate columns

```r
blocked_crossings <- blocked_crossings %>%
  mutate(Date = as.Date(`Date/Time`, format = "%m/%d/%Y")) %>%
  mutate(Year = year(Date)) %>%
  mutate(Month = month(Date)) %>%
  mutate(Time = format(as.POSIXct(`Date/Time`, format="%H:%M"), format="%H:%M"))

# as discovered in a later step, there is inconsistent capitalization, make everything uppercase for co
blocked_crossings <- blocked_crossings %>%
  mutate(City = str_to_upper(City)) %>%
  mutate(State = str_to_upper(State)) %>%
  mutate(Railroad = str_to_upper(Railroad)) %>%
  mutate(County = str_to_upper(County)) %>%
  mutate(`Crossing ID` = str_to_upper(`Crossing ID`)) %>%
  mutate(Reason = str_to_upper(Reason)) %>%
  mutate(`Immediate Impacts` = str_to_upper(`Immediate Impacts`)) %>%
  mutate(`Additional Comments` = str_to_upper(`Additional Comments`))
```

## 1.How many total complaints were filed in each year: 2022, 2023, and 2024?

```r
count(X22_blocked_crossings)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 30749
```

```r
count(X23_blocked_crossings)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 19306
```

```r
count(X24_blocked_crossings)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 26687
```

There were 30749 complaints in 2022, 19306 complaints in 2023, and 26687 complaints in 2024.

## 2. Which railroad company is associated with the most complaints?

```r
railroad_max_complaints <- blocked_crossings %>%
  group_by(Railroad) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

print(railroad_max_complaints)
```

```
## # A tibble: 380 x 2
##    Railroad Count
##    <chr>    <int>
##  1 UP       28027
##  2 NS       16368
##  3 CSX      11902
##  4 BNSF      8279
##  5 BRC       1370
##  6 GTW       1270
##  7 FEC        675
##  8 WC         526
##  9 KCS        496
## 10 NIRC       439
## # i 370 more rows
```

Union Pacific, UP, had 28027 complaints between 2022-2024.

### 3. What are the top 10 states with the highest number of blocked train crossing complaints?

```
states_most_blocked <- blocked_crossings %>%
  group_by(State) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

print(states_most_blocked)
```

```
## # A tibble: 49 x 2
##    State Count
##    <chr> <int>
##  1 TX    19100
##  2 IL     6534
##  3 OH     6103
##  4 IN     4204
##  5 TN     3401
##  6 IA     2926
##  7 MI     2840
##  8 GA     2780
##  9 KS     2609
## 10 AL     2582
## # i 39 more rows
```

Texas has the most blocked train crossings at 19,100. Illinois is second, with 6534 complaints over the three year period.

### 4. What are the top 10 cities with the highest number of blocked train crossing complaints?

```
cities_most_blocked <- blocked_crossings %>%
  group_by(City) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count)) %>%
  slice_head(n = 10)

print(cities_most_blocked)
```

```
## # A tibble: 10 x 2
##    City         Count
##    <chr>        <int>
##  1 HOUSTON      11927
##  2 NORFOLK       1334
##  3 NASHVILLE     1192
##  4 CHICAGO       1181
##  5 PERU           731
##  6 JACKSONVILLE   730
##  7 WICHITA        683
```

```
##  8 HIXSON         641
##  9 NORTHWOOD      618
## 10 BEDFORD PARK   586
```

Houston has the most reported blocked crossings, at nearly 12000 reports within a three year span.

## 5. Which crossing IDs had the most complaints?

```
# make sure city and state of crossing ID is included
crossings_most_blocked <- blocked_crossings %>%
  group_by(`Crossing ID`, City, State) %>%
  summarise(Count = n(), .groups = "drop") %>%
  #I had to drop the groups so they table would properly show up
  arrange(desc(Count)) %>%
  slice_head(n = 10)

print(crossings_most_blocked)
```

```
## # A tibble: 10 x 4
##    `Crossing ID` City         State Count
##    <chr>         <chr>        <chr> <int>
##  1 859522Y       HOUSTON      TX     1388
##  2 288226J       HOUSTON      TX      928
##  3 288224V       HOUSTON      TX      861
##  4 288221A       HOUSTON      TX      783
##  5 869221F       CHICAGO      IL      695
##  6 859523F       HOUSTON      TX      669
##  7 869223U       BEDFORD PARK IL      572
##  8 288227R       HOUSTON      TX      517
##  9 859524M       HOUSTON      TX      504
## 10 509423L       NORTHWOOD    OH      480
```

Houston, Texas has seven of the ten most reported blocked crossings in the country. Crossing ID 859522Y had 1388 reports in 3 years.

(It could be interesting to dig more into this crossing in Houston for a story)

## 6. Which states have seen the greatest increase in complaints from 2022 to 2024?

```
increase_in_complaints <- blocked_crossings %>%
  group_by(State, Year) %>%
  summarise(Count = n(), .groups = "drop") %>%
  pivot_wider(names_from = Year, values_from = Count) %>%
  mutate(Increase = `2024` - `2022`) %>%
  mutate(Percent_Increase = (Increase / `2022`) * 100) %>%
  arrange(desc(Increase))

print(increase_in_complaints)
```

```
## # A tibble: 49 x 6
##    State `2022` `2023` `2024` Increase Percent_Increase
##    <chr>  <int>  <int>  <int>    <int>            <dbl>
##  1 TX      6508   4322   8270     1762             27.1
##  2 VA       524    518   1484      960            183.
##  3 CA       579    420   1053      474             81.9
##  4 KS       903    525   1181      278             30.8
##  5 OK       301    427    560      259             86.0
##  6 CO       415    517    577      162             39.0
##  7 OR       203    179    299       96             47.3
##  8 MD        52    163    125       73            140.
##  9 KY       388    302    459       71             18.3
## 10 LA       323    339    389       66             20.4
## # i 39 more rows
```

```
#this is a bad metric, since the crossings with less complaints that got more with time will show super
percent_increase_in_complaints <- increase_in_complaints %>%
  arrange(desc(Percent_Increase))

print(percent_increase_in_complaints)
```

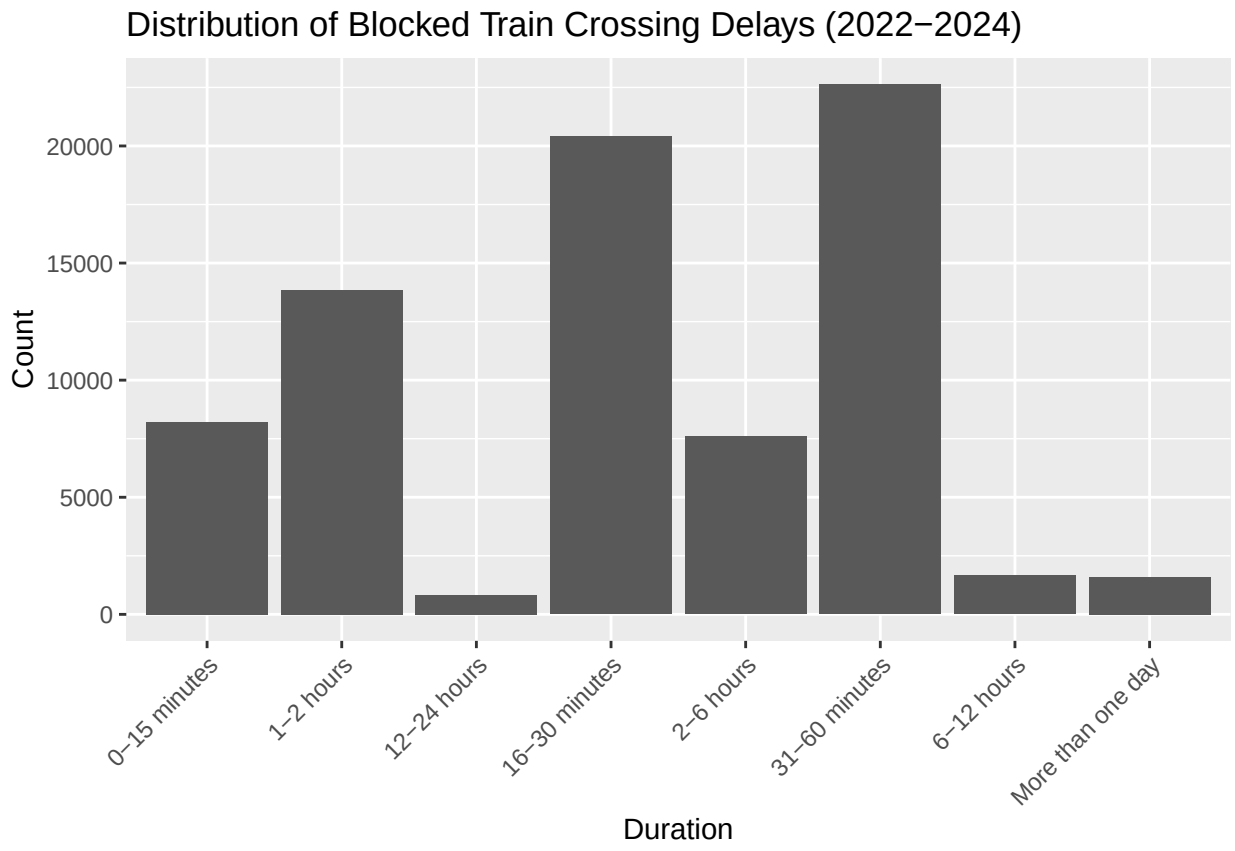
```
## # A tibble: 49 x 6
##    State `2022` `2023` `2024` Increase Percent_Increase
##    <chr>  <int>  <int>  <int>    <int>            <dbl>
##  1 AK         1      2      5        4            400
##  2 VA       524    518   1484      960            183.
##  3 SD         8      9     22       14            175
##  4 MD        52    163    125       73            140.
##  5 NV        18     43     43       25            139.
##  6 NM        51     26    107       56            110.
##  7 OK       301    427    560      259             86.0
##  8 CA       579    420   1053      474             81.9
##  9 ME         3     10      5        2             66.7
## 10 MT        86    135    139       53             61.6
## # i 39 more rows
```

Texas has seen the greatest total increase in blocked train crossing complaints from 2022 to 2024, with 1,762 more complaints (+27.1%). It's followed by Virginia, which saw an increase of 960 complaints (+183.2%), and California with 474 more complaints (+81.9%) over the same period.

## 7. What is the distribution of blocked train crossing delays? What is the quantity/percentage over 15 minutes?

```
delay_distribution <- blocked_crossings %>%
  group_by(Duration) %>%
  summarise(Count = n()) %>%
  mutate(Percentage = (Count / sum(Count)) * 100) %>%
  arrange(desc(Count))

ggplot(delay_distribution, aes(x = Duration, y = Count)) +
  geom_bar(stat = "identity") +
```

```
  labs(title = "Distribution of Blocked Train Crossing Delays (2022-2024)",
       x = "Duration",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
delays_over_fifteen <- blocked_crossings %>%
  filter(Duration != "0-15 minutes")

percent_delays_over_fifteen <- nrow(delays_over_fifteen)/nrow(blocked_crossings) * 100

print(percent_delays_over_fifteen)
```

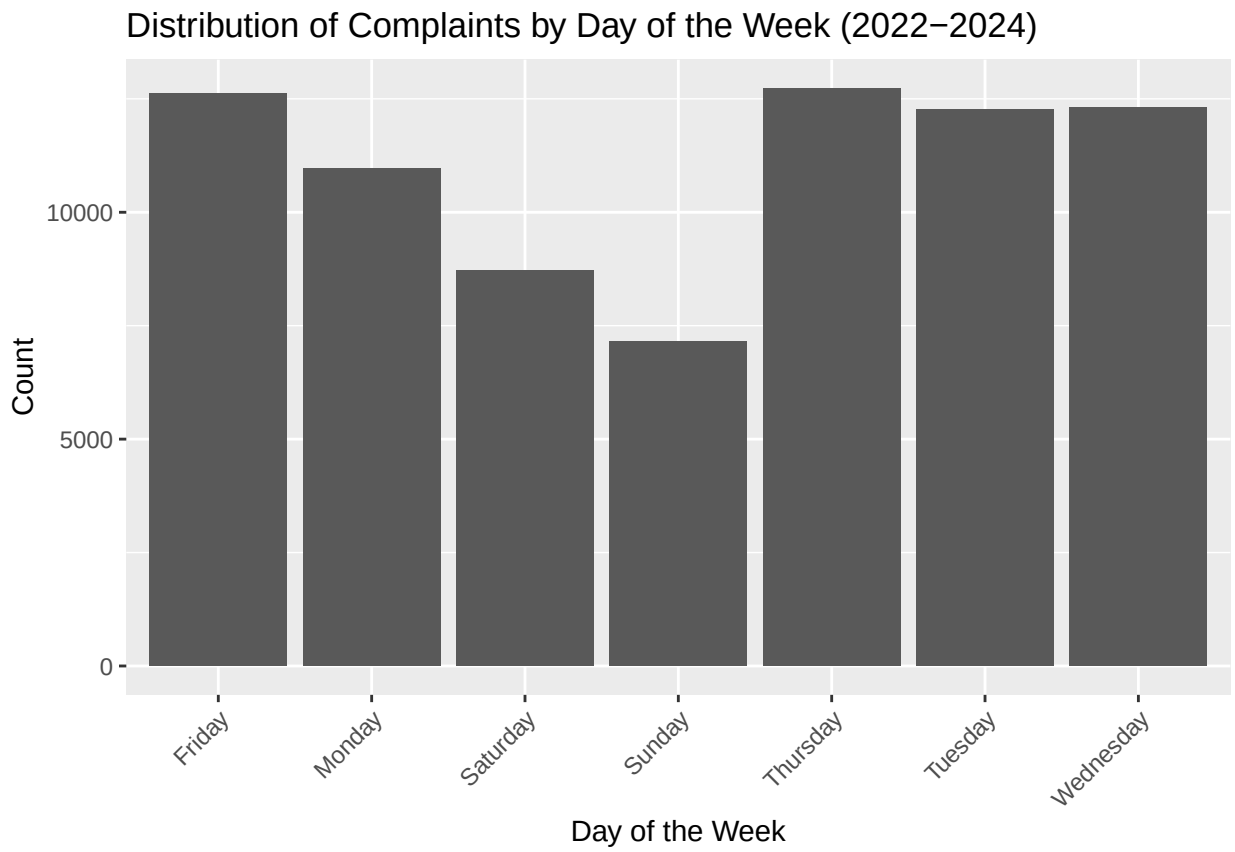
```
## [1] 89.31485
```

89% of reported delays were over 15 minutes.

## 8. What is the distribution of complaints by day of the week?

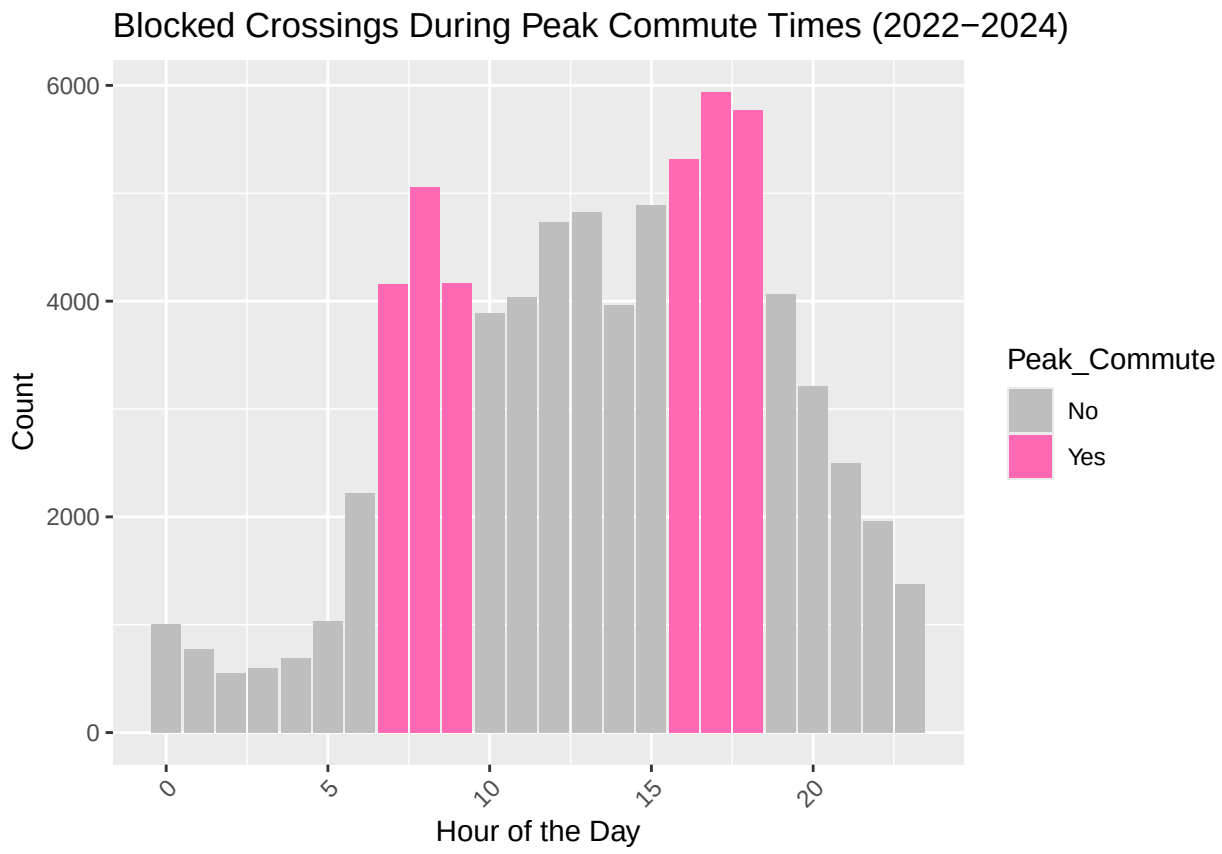
```
dow_complaints <- blocked_crossings %>%
  mutate(Day_of_Week = weekdays(Date)) %>%
  group_by(Day_of_Week) %>%
  summarise(Count = n()) %>%
```

```r
  arrange(desc(Count))

ggplot(dow_complaints, aes(x = Day_of_Week, y = Count)) +
  geom_bar(stat = "identity") +
  labs(title = "Distribution of Complaints by Day of the Week (2022-2024)",
       x = "Day of the Week",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




### 9. How many blocked crossings were reported during peak commute times?

```r
#let's do this by mapping what the distribution is for every single hour during the day
blocked_crossings <- blocked_crossings %>%
  mutate(Hour = hour(`Date/Time`)) %>%
  mutate(Peak_Commute = ifelse(Hour >= 7 & Hour <= 9 | Hour >= 16 & Hour <= 18, "Yes", "No"))

ggplot(blocked_crossings, aes(x = Hour)) +
  geom_bar(aes(fill = Peak_Commute), position = "dodge") +
  labs(title = "Blocked Crossings During Peak Commute Times (2022-2024)",
       x = "Hour of the Day",
       y = "Count") +
  scale_fill_manual(values = c("Yes" = "hotpink", "No" = "grey")) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




## 10. What day had the most train crossing complaints nationwide?

```
# find what day had the most complaints nationwide
day_most_complaints <- blocked_crossings %>%
  group_by(Date) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count)) %>%
  slice_head(n = 1)

print(day_most_complaints)
```

```
## # A tibble: 1 x 2
##   Date       Count
##   <date>     <int>
## 1 2022-05-20   243
```

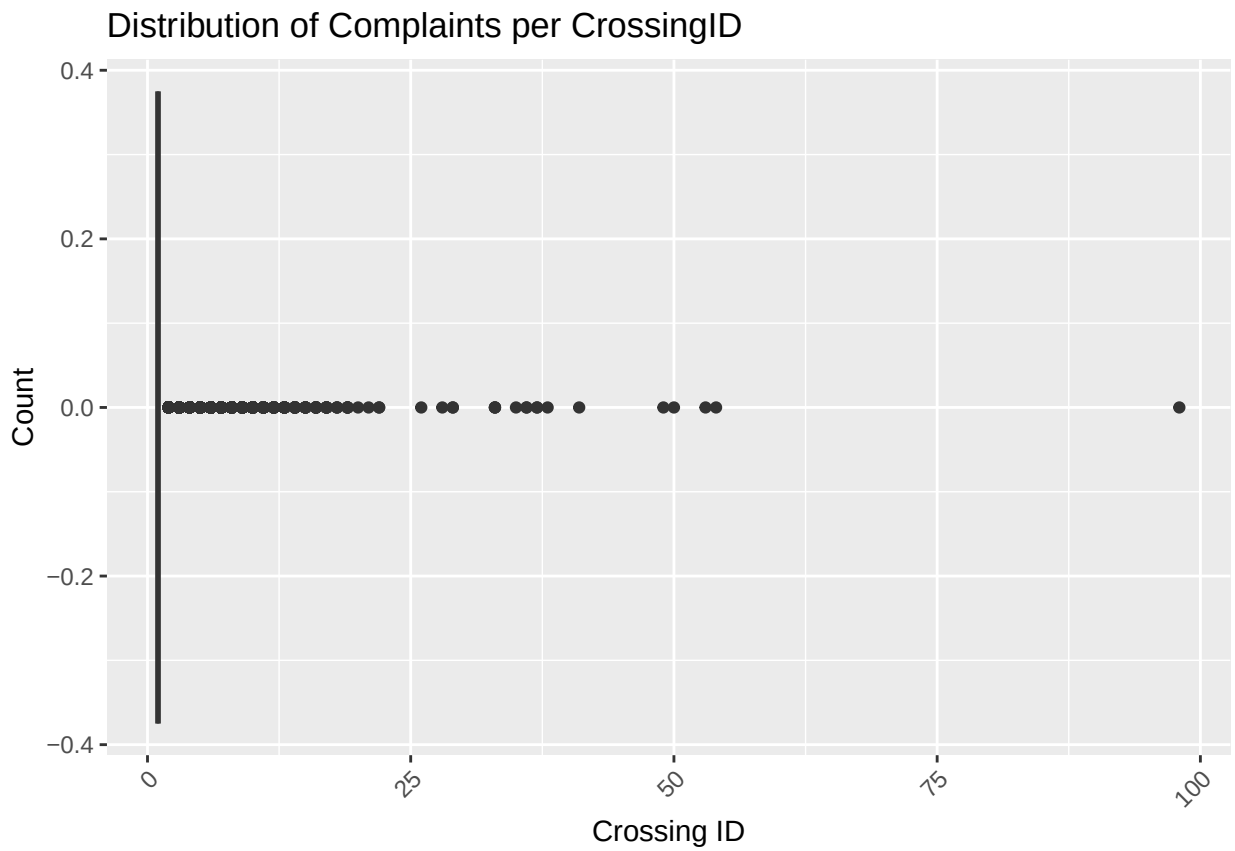
Nationwide May 20, 2022 saw the most blocked crossing complaints at 243 total complaints.

## 11. What specific crossing had the most complaints in a single day?

```r
# find what specific crossing had the most complaints in a single day
top_complaint_day_crossing <- blocked_crossings %>%
  group_by(`Crossing ID`, Date, City, State) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))

print(top_complaint_day_crossing)
```

```
## # A tibble: 56,968 x 5
##    `Crossing ID` Date       City           State Count
##    <chr>         <date>     <chr>          <chr> <int>
##  1 859523F       2023-08-18 HOUSTON        TX       98
##  2 760696R       2024-11-13 CABAZON        CA       54
##  3 760697X       2024-11-13 CABAZON        CA       53
##  4 254790K       2024-07-30 SALT LAKE CITY UT       50
##  5 629771V       2024-05-20 ROCKY MOUNT    NC       49
##  6 869223U       2022-06-09 BEDFORD PARK   IL       41
##  7 352253E       2022-03-13 HELENA         AL       38
##  8 263984P       2022-04-07 OWEGO          NY       37
##  9 288226J       2022-01-24 HOUSTON        TX       37
## 10 092504C       2023-02-18 NAPAVINE       WA       36
## # i 56,958 more rows
```

```r
# based on the results from top_complaint_day_crossing, let's look to see what's the distribution of th
ggplot(top_complaint_day_crossing, aes(x = Count)) +
  geom_boxplot() +
  labs(title = "Distribution of Complaints per CrossingID",
       x = "Crossing ID",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# lowkey a useless graph, except for showing that most crossing complaints were like 1 per incident
```

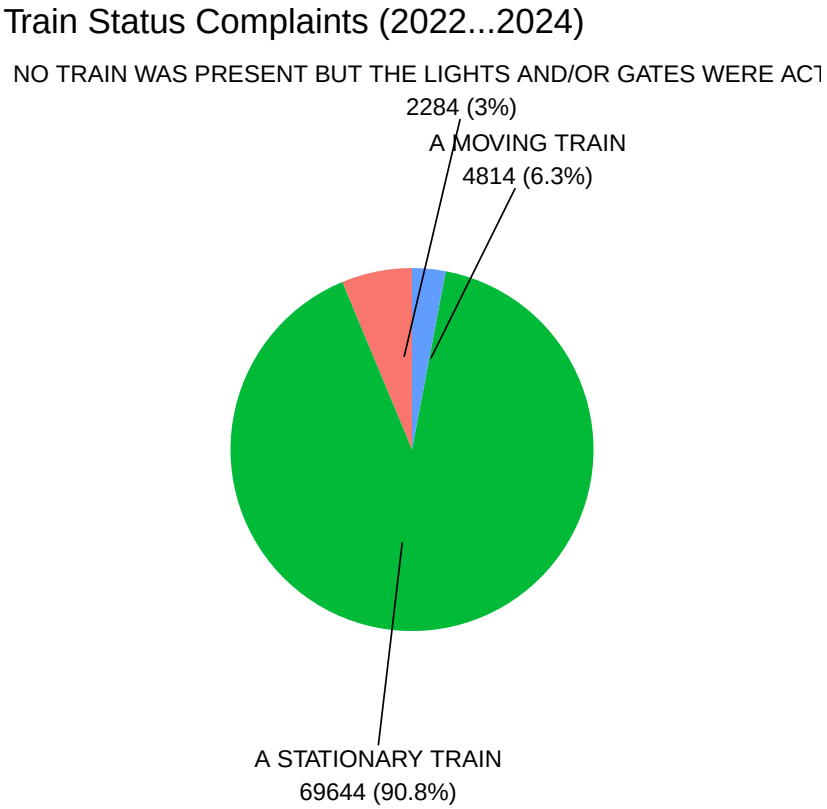
Crossing ID 859523F in Houston, TX saw the most complaints in a single day at a single crossing at 98 total complaints.

### 12. What is the split between complaints of moving, stationary trains, or no train was present but the gate was activated?

```
#asked chatgpt for styling help, hence ggrepel
train_status <- blocked_crossings %>%
  count(Reason) %>%
  mutate(
    percent = round(n / sum(n) * 100, 1),
    label = paste0(Reason, "\n", n, " (", percent, "%)"),
    ypos = cumsum(n) - 0.5 * n  # position for label
  )

plot_train_status <- train_status %>%
  ggplot(aes(x = "", y = n, fill = Reason)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y") +
  geom_text_repel(
    aes(y = ypos, label = label),
```

```
    nudge_x = 1.3,
    show.legend = FALSE,
    segment.size = 0.3,
    size = 3
  ) +
  labs(
    title = "Train Status Complaints (2022-2024)",
    fill = "Reason"
  ) +
  theme_void() +
  theme(legend.position = "none")  # << this hides the sidebar legend

print(plot_train_status)
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <93>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <e2>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <80>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Train Status Complaints (2022-2024)' in 'mbcsToSbcs':
## dot substituted for <93>
```




### 13. How many complaints list no train present but still report gates down or signals activated?

```
train_status %>%
  filter(str_detect(Reason, "NO TRAIN WAS PRESENT")) %>%
  arrange(desc(n))
```

```
## # A tibble: 1 x 5
##   Reason                                                  n percent label  ypos
##   <chr>                                               <int>   <dbl> <chr> <dbl>
## 1 NO TRAIN WAS PRESENT BUT THE LIGHTS AND/OR GATES WE~  2284       3 "NO ~ 75600
```

2284 complaints listed the reason for blockage as "no train was present but the lights and/or gates were activated".

### 14. How many complaints mentioned "first responders were observed being unable to cross the tracks?"

```
responders_blocked <- blocked_crossings %>%
  filter(str_detect(`Immediate Impacts`, "FIRST RESPONDERS"))

unique(blocked_crossings$`Immediate Impacts`)
```

```
## [1] NA
## [2] "FIRST RESPONDERS WERE OBSERVED BEING UNABLE TO CROSS THE TRACKS"
## [3] "PEDESTRIANS WERE OBSERVED CLIMBING ON, OVER, OR THROUGH THE TRAIN CARS"
## [4] "FIRST RESPONDERS WERE OBSERVED BEING UNABLE TO CROSS THE TRACKS, PEDESTRIANS WERE OBSERVED CLIM
```

```
print(responders_blocked)
```

```
## # A tibble: 15,416 x 17
##    `Crossing ID` City  State Street County Railroad `Date/Time`         Duration
##    <chr>         <chr> <chr> <chr>  <chr>  <chr>    <dttm>              <chr>
##  1 304225W       MOBI~ AL    DAUPH~ MOBILE ALE      2022-07-13 21:10:00 0-15 mi~
##  2 306365F       TUSC~ AL    6TH A~ TUSCA~ ABS      2022-01-22 07:06:52 0-15 mi~
##  3 350272C       LIND~ AL    RAILR~ MAREN~ MNBR     2022-01-04 15:48:00 31-60 m~
##  4 351419D       MOBI~ AL    PEDES~ MOBILE CSX      2022-08-27 17:36:00 2-6 hou~
##  5 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-05-04 09:02:00 31-60 m~
##  6 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-05-04 08:10:00 1-2 hou~
##  7 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-04-29 17:46:00 2-6 hou~
##  8 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-03-13 20:43:02 31-60 m~
##  9 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-03-13 20:03:00 1-2 hou~
## 10 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-03-03 20:33:00 31-60 m~
## # i 15,406 more rows
## # i 9 more variables: Reason <chr>, `Immediate Impacts` <chr>,
## #   `Additional Comments` <chr>, Date <date>, Year <dbl>, Month <dbl>,
## #   Time <chr>, Hour <int>, Peak_Commute <chr>
```

```
nrow(responders_blocked)
```

```
## [1] 15416
```

Over 15000 complaints mentioned that first responders were blocked by the train crossing.

### 15. How many complaints mentioned "pedestrians were observed climbing on, over, or through train cars?"

```
pedestrians_crossings <- blocked_crossings %>%
  filter(str_detect(`Immediate Impacts`, "PEDESTRIANS"))

print(pedestrians_crossings)
```

```
## # A tibble: 17,974 x 17
##    `Crossing ID` City  State Street County Railroad `Date/Time`         Duration
##    <chr>         <chr> <chr> <chr>  <chr>  <chr>    <dttm>              <chr>
##  1 306362K       TUSC~ AL    12TH ~ TUSCA~ ABS      2022-07-11 14:11:00 2-6 hou~
##  2 306363S       TUSC~ AL    10TH ~ TUSCA~ ABS      2022-08-11 17:00:00 1-2 hou~
##  3 306363S       TUSC~ AL    10TH ~ TUSCA~ ABS      2022-06-29 09:27:00 31-60 m~
##  4 351251M       GREE~ AL    OLD S~ BUTLER CSX      2022-07-24 10:36:00 1-2 hou~
##  5 351378B       ATMO~ AL    JAMES~ ESCAM~ CSX      2022-07-18 12:54:00 2-6 hou~
##  6 351379H       ATMO~ AL    JAMES~ ESCAM~ CSX      2022-07-18 12:53:00 6-12 ho~
##  7 351419D       MOBI~ AL    PEDES~ MOBILE CSX      2022-09-15 11:26:00 31-60 m~
```

```
##  8 351419D       MOBI~ AL    PEDES~ MOBILE CSX      2022-08-27 17:36:00 2-6 hou~
##  9 351419D       MOBI~ AL    PEDES~ MOBILE CSX      2022-08-13 18:00:00 31-60 m~
## 10 351419D       MOBI~ AL    PEDES~ MOBILE CSX      2022-08-13 18:00:00 31-60 m~
## # i 17,964 more rows
## # i 9 more variables: Reason <chr>, `Immediate Impacts` <chr>,
## #   `Additional Comments` <chr>, Date <date>, Year <dbl>, Month <dbl>,
## #   Time <chr>, Hour <int>, Peak_Commute <chr>
```

```
#percentage of citings that mention pedestrians
percent_pedestrians <- nrow(pedestrians_crossings)/nrow(blocked_crossings) * 100

print(percent_pedestrians)
```

```
## [1] 23.42133
```

Almost 18,000 complaints, 23% of all complaints, cited pedestrians climbing over, under, or through the blocked train crossing.

## 16. What percentage of comments cited "traffic"?

```
#using string detect method I learned in PLAN372
mentioned_traffic <- blocked_crossings %>%
  filter(str_detect(`Additional Comments`, regex("Traffic", ignore_case = TRUE)))

print(mentioned_traffic)
```

```
## # A tibble: 7,553 x 17
##    `Crossing ID` City  State Street County Railroad `Date/Time`         Duration
##    <chr>         <chr> <chr> <chr>  <chr>  <chr>    <dttm>              <chr>
##  1 304225W       MOBI~ AL    DAUPH~ MOBILE ALE      2022-09-08 17:50:58 16-30 m~
##  2 304225W       MOBI~ AL    DAUPH~ MOBILE ALE      2022-07-13 21:10:00 0-15 mi~
##  3 304230T       MOBI~ AL    SPRIN~ MOBILE ALE      2022-09-08 18:15:00 0-15 mi~
##  4 304230T       MOBI~ AL    SPRIN~ MOBILE ALE      2022-09-08 12:00:00 16-30 m~
##  5 304231A       MOBI~ AL    MOFFE~ MOBILE ALE      2022-03-17 10:10:00 16-30 m~
##  6 306362K       TUSC~ AL    12TH ~ TUSCA~ ABS      2022-07-11 14:11:00 2-6 hou~
##  7 306363S       TUSC~ AL    10TH ~ TUSCA~ ABS      2022-06-29 09:27:00 31-60 m~
##  8 351429J       MOBI~ AL    PILLA~ MOBILE CSX      2022-02-21 19:32:00 31-60 m~
##  9 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-05-04 08:10:00 1-2 hou~
## 10 352253E       HELE~ AL    MAIN ~ SHELBY CSX      2022-05-04 07:40:00 31-60 m~
## # i 7,543 more rows
## # i 9 more variables: Reason <chr>, `Immediate Impacts` <chr>,
## #   `Additional Comments` <chr>, Date <date>, Year <dbl>, Month <dbl>,
## #   Time <chr>, Hour <int>, Peak_Commute <chr>
```

```
percent_mentioned_traffic <- nrow(mentioned_traffic)/nrow(blocked_crossings) * 100

print(percent_mentioned_traffic)
```

```
## [1] 9.842068
```

Nearly 10% of complaints explicitly mentioned "traffic" in their additional comments about the incident, totalling over 7,500 complaints from 2022-2024.

### 17. Which incidents received the most complaints? Which date/time and crossing ID combination appears most frequently in the dataset?

```
incidents_most_complaints <- blocked_crossings %>%
  mutate(rounded_time = floor_date(`Date/Time`, unit = "hour")) %>%
  group_by(rounded_time, `Crossing ID`, City, State, Railroad) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count)) %>%
  slice_head(n = 10)

incidents_most_complaints
```

```
## # A tibble: 10 x 6
##    rounded_time        `Crossing ID` City           State Railroad Count
##    <dttm>              <chr>         <chr>          <chr> <chr>    <int>
##  1 2023-08-18 18:00:00 859523F       HOUSTON        TX    UP          94
##  2 2024-05-20 15:00:00 629771V       ROCKY MOUNT    NC    CSX         49
##  3 2024-07-30 19:00:00 254790K       SALT LAKE CITY UT    UFRC        43
##  4 2022-06-09 10:00:00 869223U       BEDFORD PARK   IL    BRC         41
##  5 2023-02-18 03:00:00 092504C       NAPAVINE       WA    BNSF        36
##  6 2022-03-13 20:00:00 352253E       HELENA         AL    CSX         33
##  7 2022-06-09 10:00:00 869221F       CHICAGO        IL    BRC         33
##  8 2024-08-30 13:00:00 139521E       BETHESDA       MD    MONZ        33
##  9 2024-11-13 18:00:00 760696R       CABAZON        CA    UP          28
## 10 2022-01-24 18:00:00 288226J       HOUSTON        TX    UP          27
```

```
## I NEED TO DOUBLE CHECK THIS, is date and time too rigid, should it be a range?

## THIS FEELS APPROXIMATE, I would not use this in an actual story, more as a way to find interesting b
```

See comment above. Would not include this analysis in a news story since it doesn't feel precise enough.

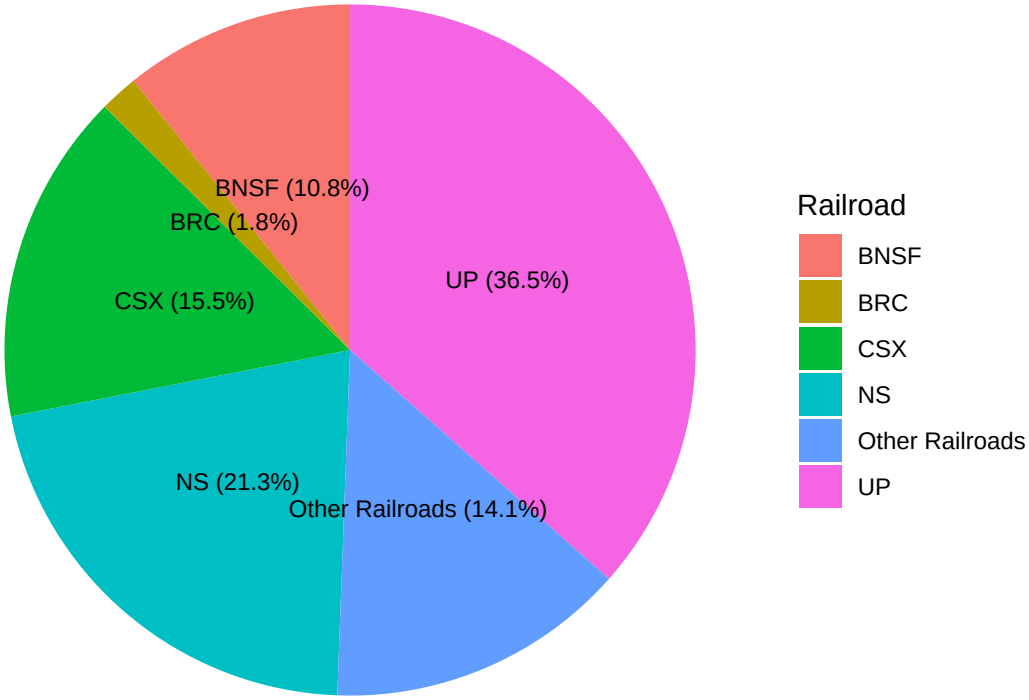
### 18. How many complaints were filed per railroad company

```
complaints_per_railroad_company <- blocked_crossings %>%
  group_by(Railroad) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

complaints_per_railroad_company
```

```
## # A tibble: 380 x 2
##    Railroad Count
##    <chr>    <int>
##  1 UP       28027
##  2 NS       16368
##  3 CSX      11902
##  4 BNSF      8279
##  5 BRC       1370
```

```
##  6 GTW       1270
##  7 FEC        675
##  8 WC         526
##  9 KCS        496
## 10 NIRC       439
## # i 370 more rows
```

```r
# identify top 5 railroads with most complaints
top_railroads <- complaints_per_railroad_company %>%
  slice_head(n = 5) %>%
  pull(Railroad)

#asked chatgpt with help making pie chart
blocked_crossings_pie <- blocked_crossings %>%
  mutate(railroad_group = ifelse(Railroad %in% top_railroads,
                                 Railroad, "Other Railroads")) %>%
  count(railroad_group, name = "Count") %>%
  arrange(desc(Count)) %>%
  mutate(perc = round(Count / sum(Count) * 100, 1),
         label = paste0(railroad_group, " (", perc, "%)"))

ggplot(blocked_crossings_pie, aes(x = "", y = Count, fill = railroad_group)) +
  geom_col(width = 1) +
  coord_polar(theta = "y") +
  theme_void() +
  guides(fill = guide_legend(title = "Railroad")) +
  geom_text(aes(label = ifelse(Count > 0, label, "")),
            position = position_stack(vjust = 0.5),
            size = 3)
```




### 19. What is the average complaint duration by company?

```
# I need to double check that since duration is a character string, it's in the appropriate order
  # answer:chatgpt suggested using ordered factors to make sure durations are in correct order
duration_factors <- c("0-15 minutes",
                      "16-30 minutes",
                      "1-2 hours",
                      "2-6 hours",
                      "6-12 hours",
                      "12-24 hours",
                      "More than one day")

# keeping it to the top 5 so I know the sample is appropriately large
duration_per_company <- blocked_crossings %>%
  mutate(Duration = factor(Duration, levels = duration_factors, ordered = TRUE)) %>%
  filter(Railroad %in% top_railroads) %>%
  group_by(Railroad, Duration) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))

duration_per_company
```

```
## # A tibble: 37 x 3
```

```
##    Railroad Duration      Count
##    <chr>    <ord>         <int>
##  1 UP       <NA>           8265
##  2 UP       16-30 minutes  6956
##  3 UP       1-2 hours      5951
##  4 NS       <NA>           3906
##  5 CSX      <NA>           3594
##  6 NS       16-30 minutes  3554
##  7 CSX      16-30 minutes  3353
##  8 UP       2-6 hours      3293
##  9 NS       1-2 hours      3184
## 10 BNSF     <NA>           2686
## # i 27 more rows
```

```
# I need to double check that since duration is a character string, it's in the appropriate order
  # answer:chatgpt suggested using ordered factors to make sure durations are in correct order
unique(blocked_crossings$Duration)
```

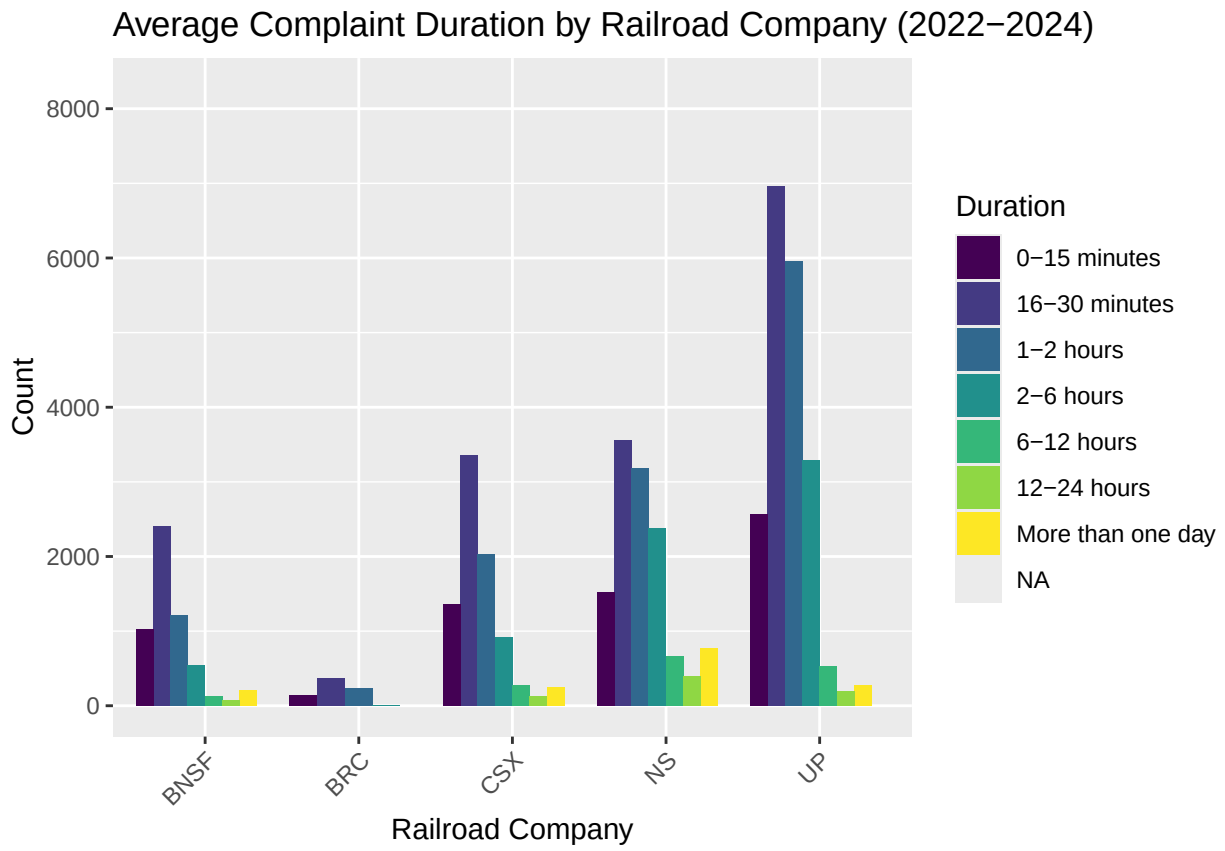
```
## [1] "0-15 minutes"      "16-30 minutes"     "2-6 hours"
## [4] "1-2 hours"         "31-60 minutes"     "6-12 hours"
## [7] "More than one day" "12-24 hours"
```

```
duration_factors <- c("0-15 minutes",
                      "16-30 minutes",
                      "1-2 hours",
                      "2-6 hours",
                      "6-12 hours",
                      "12-24 hours",
                      "More than one day")


# make sure to use median, since I'm working with ranges
# Median duration as a category (not numeric)
median_duration_per_company <- blocked_crossings %>%
  filter(Railroad %in% top_railroads, !is.na(Duration)) %>%
  group_by(Railroad, Duration) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(Railroad, Duration) %>%
  group_by(Railroad) %>%
  mutate(CumSum = cumsum(Count), Total = sum(Count)) %>%
  filter(CumSum >= Total / 2) %>%
  slice_head(n = 1) %>%
  select(Railroad, Median_Duration = Duration)

median_duration_per_company
```

```
## # A tibble: 5 x 2
## # Groups:   Railroad [5]
##   Railroad Median_Duration
##   <chr>    <chr>
## 1 BNSF     16-30 minutes
## 2 BRC      16-30 minutes
## 3 CSX      16-30 minutes
```

```
## 4 NS       16-30 minutes
## 5 UP       16-30 minutes
```

```r
#huh, returning all 16-30 minutes, let's make bar charts for all the 5 to see average delay
ggplot(duration_per_company, aes(x = Railroad, y = Count, fill = Duration)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Complaint Duration by Railroad Company (2022-2024)",
       x = "Railroad Company",
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




The top 5 railroad companies by complaint volume had an average reported blocked crossing time between 16-30 minutes.

### 20. Which counties had the highest number of complaints across all three years?

```r
complaint_counties <- blocked_crossings %>%
  group_by(County, State) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count)) %>%
  slice_head(n = 10)

print(complaint_counties)
```

```
## # A tibble: 10 x 3
##    County   State Count
##    <chr>    <chr> <int>
##  1 HARRIS   TX    12831
##  2 COOK     IL     3723
##  3 NORFOLK  VA     1333
##  4 DAVIDSON TN     1082
##  5 SHELBY   AL      950
##  6 WOOD     OH      950
##  7 WAYNE    MI      911
##  8 HAMILTON TN      894
##  9 TARRANT  TX      867
## 10 BUTLER   OH      848
```

```
#oh no! there are different reports based on capitalization! I need to go back through and clean up blo
```

Harris County, Texas received the most complaints between 2022-2024, totalling over 12,800 reports of blocked crossings. Cook County, IL lagged far behind in second place with 3723 reported blocked crossings.

## 21. How many complaints were filed in each month across the dataset?

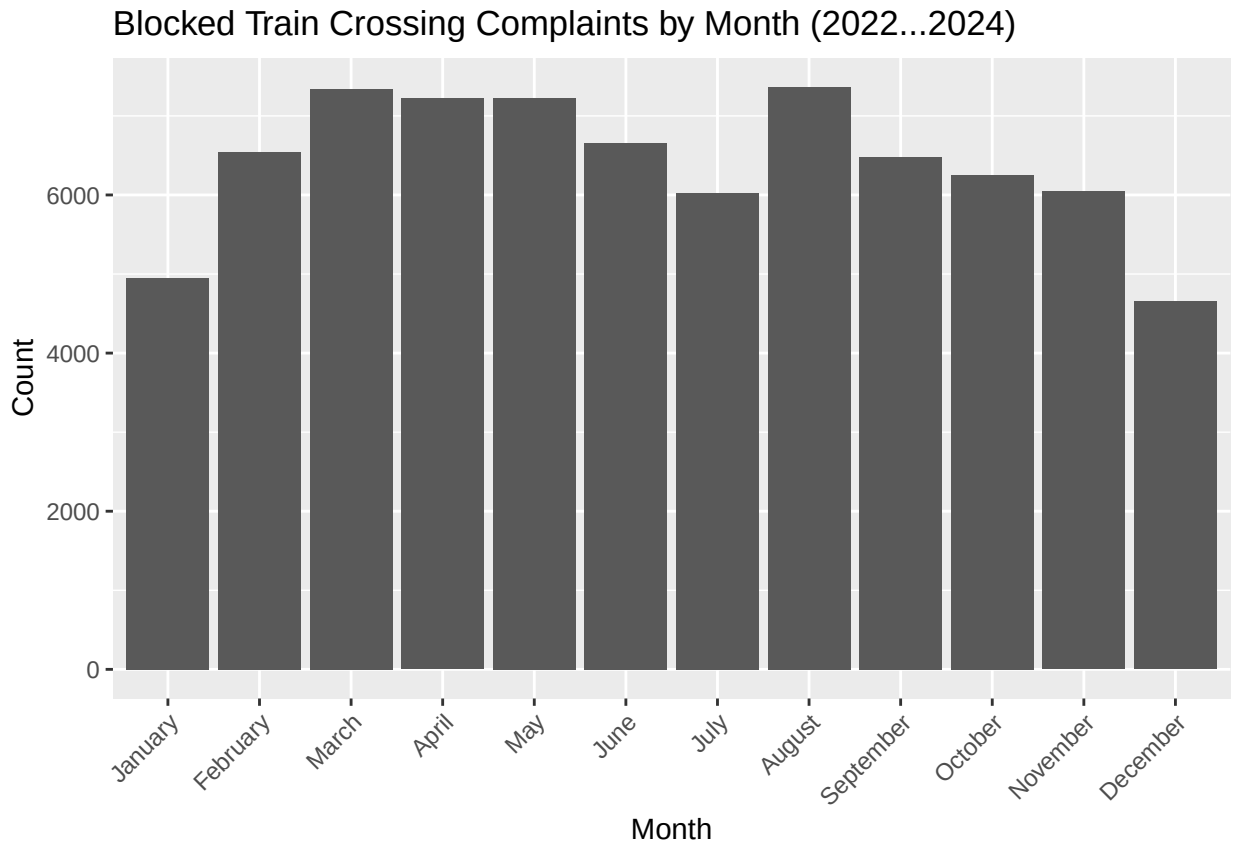
```
blocked_crossings <- blocked_crossings %>%
  mutate(Month = factor(month.name[as.numeric(Month)], levels = month.name, ordered = TRUE))

complaints_per_month <- blocked_crossings %>%
  group_by(Month) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(Month)

print(complaints_per_month)
```

```
## # A tibble: 12 x 2
##    Month     Count
##    <ord>     <int>
##  1 January    4946
##  2 February   6540
##  3 March      7336
##  4 April      7218
##  5 May        7225
##  6 June       6655
##  7 July       6027
##  8 August     7367
##  9 September  6482
## 10 October    6250
## 11 November   6042
## 12 December   4654
```

```
ggplot(complaints_per_month, aes(x = Month, y = Count)) +
  geom_bar(stat = "identity") +
  labs(title = "Blocked Train Crossing Complaints by Month (2022-2024)",
       x = "Month",
```

```
       y = "Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <93>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Blocked Train Crossing Complaints by Month (2022-2024)'
## in 'mbcsToSbcs': dot substituted for <93>
```



```r
mean(complaints_per_month$Count)
```

```
## [1] 6395.167
```

August had the most complaints per month from 2022-2024, totalling nearly 7400. The mean complaints per month from that period is just under 6400.

### 22. How many total crossing locations appear only once in the dataset?

```r
one_hit_wonders <- blocked_crossings %>%
  group_by(`Crossing ID`, City, State) %>%
  summarise(Count = n(), .groups = "drop") %>%
  filter(Count == 1)

one_hit_wonders
```

```
## # A tibble: 7,163 x 4
##    `Crossing ID` City          State Count
##    <chr>         <chr>         <chr> <int>
##  1 000112Y       TOLEDO        OH        1
##  2 000126G       SAMARIA       MI        1
##  3 000139H       IDA           MI        1
##  4 000172H       MILAN         MI        1
##  5 000247E       WHITMORE LAKE MI        1
##  6 000253H       WHITMORE LAKE MI        1
##  7 000261A       HAMBURG       MI        1
##  8 000288J       HOWELL        MI        1
##  9 000292Y       HOWELL        MI        1
## 10 000356H       OWOSSO        MI        1
## # i 7,153 more rows
```

```r
nrow(one_hit_wonders)/nrow(blocked_crossings)
```

```
## [1] 0.09333872
```

About 10% of crossings in the dataset received only one complaint in the three year span.

## North Carolina specific questions

```r
# start by filtering the data for just NC
nc_blocked_crossings <- blocked_crossings %>%
  filter(State == "NC")

nc_blocked_crossings
```

```
## # A tibble: 411 x 17
##    `Crossing ID` City  State Street County Railroad `Date/Time`         Duration
##    <chr>         <chr> <chr> <chr>  <chr>  <chr>    <dttm>              <chr>
##  1 465809H       FUQU~ NC    WILBU~ WAKE   NS       2022-03-03 12:01:00 16-30 m~
##  2 465814E       FUQU~ NC    JUDD ~ WAKE   NS       2022-03-03 12:00:00 16-30 m~
##  3 465892L       WADE  NC    PRIVA~ CUMBE~ NS       2022-03-09 13:08:00 31-60 m~
##  4 465990C       ROBB~ NC    GREEN~ MOORE  ACWR     2022-11-06 13:00:00 16-30 m~
##  5 466258F       EAGL~ NC    SAMAR~ MOORE  ACWR     2022-06-19 21:09:00 2-6 hou~
##  6 470210K       WALK~ NC    WILLI~ FORSY~ NS       2022-05-11 07:19:27 1-2 hou~
##  7 470211S       WALK~ NC    HAMMO~ FORSY~ NS       2022-05-11 07:19:12 1-2 hou~
##  8 470212Y       WALK~ NC    NORTH~ FORSY~ NS       2022-05-11 07:18:54 1-2 hou~
##  9 470213F       WALK~ NC    OAK R~ FORSY~ NS       2022-08-30 10:02:00 1-2 hou~
## 10 470213F       WALK~ NC    OAK R~ FORSY~ NS       2022-05-11 07:17:46 31-60 m~
## # i 401 more rows
## # i 9 more variables: Reason <chr>, `Immediate Impacts` <chr>,
## #   `Additional Comments` <chr>, Date <date>, Year <dbl>, Month <ord>,
## #   Time <chr>, Hour <int>, Peak_Commute <chr>
```

### 23. How many complaints were filed in North Carolina in total from 2022 to 2024?

```
nc_total_complaints <- nrow(nc_blocked_crossings)

print(nc_total_complaints)
```

```
## [1] 411
```

There were 411 complaints of blocked train crossings in North Carolina between 2022-2024.

### 24. Which city in North Carolina had the most blocked train crossing complaints?

```
nc_city_top_blockings <- nc_blocked_crossings %>%
  group_by(City) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

nc_city_top_blockings
```

```
## # A tibble: 69 x 2
##    City          Count
##    <chr>         <int>
##  1 ROCKY MOUNT     176
##  2 CHARLOTTE        78
##  3 FAYETTEVILLE     26
##  4 SHARPSBURG       17
##  5 DURHAM           11
##  6 BOSTIC            5
```

```
##  7 WALKERTOWN        5
##  8 FUQUAY-VARINA     4
##  9 GREENSBORO        4
## 10 NEW BERN          4
## # i 59 more rows
```

Rocky Mount, NC had the most reporting blocked train crossings, at 176 in the 3 year span.

## 25. What county in North Carolina had the most blocked train crossing complaints?

```
nc_county_top_blockings <- nc_blocked_crossings %>%
  group_by(County, City) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))
```

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

```
nc_county_top_blockings
```

```
## # A tibble: 72 x 3
## # Groups:   County [47]
##    County      City          Count
##    <chr>       <chr>         <int>
##  1 NASH        ROCKY MOUNT     174
##  2 MECKLENBURG CHARLOTTE        78
##  3 CUMBERLAND  FAYETTEVILLE     26
##  4 NASH        SHARPSBURG       17
##  5 DURHAM      DURHAM           11
##  6 FORSYTH     WALKERTOWN        5
##  7 RUTHERFORD  BOSTIC            5
##  8 CRAVEN      NEW BERN          4
##  9 GUILFORD    GREENSBORO        4
## 10 WAKE        FUQUAY-VARINA     4
## # i 62 more rows
```

Nash County, home to Rocky Mount, had the most reported blocked crossings.

## 26. What are the most frequently reported railroad companies in North Carolina complaints?

```
nc_top_company_blockings <- nc_blocked_crossings %>%
  group_by(Railroad) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

print(nc_top_company_blockings)
```

```
## # A tibble: 13 x 2
##    Railroad Count
##    <chr>    <int>
##  1 CSX        282
##  2 NS         105
##  3 ACWR         9
##  4 WSS          3
##  5 CA           2
##  6 CER          2
##  7 CLNA         2
##  8 GSM          1
##  9 LRS          1
## 10 NCVA         1
## 11 RJCS         1
## 12 TBRY         1
## 13 YVRR         1
```

```
#I wonder what the total percent of complaints CSX had, seems about 70% of all NC complaints if I had t
```

### 27. What percent of total complaints in NC was CSX responsible for?

```
csx_blocked_percent_nc <- nc_top_company_blockings %>%
  filter(Railroad == "CSX") %>%
  summarise(Percent = (Count / nc_total_complaints) * 100)

print(csx_blocked_percent_nc)
```

```
## # A tibble: 1 x 1
##   Percent
##     <dbl>
## 1    68.6
```

CSX had 282 reported complaints from 2022-2024, accounting for 69% of all complaints in North Carolina over the three year span.

### 28. Which crossing IDs in North Carolina were reported the most frequently?

```
# based on other questions asked earlier, this might be redundant, I'm going to guess it's in Rocky Mou
nc_crossings_most_blocked <- nc_blocked_crossings %>%
  group_by(`Crossing ID`, City, County) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))

print(nc_crossings_most_blocked)
```

```
## # A tibble: 134 x 4
##    `Crossing ID` City         County      Count
##    <chr>         <chr>        <chr>       <int>
```

```
##  1 629771V       ROCKY MOUNT  NASH          167
##  2 716178E       CHARLOTTE    MECKLENBURG    19
##  3 629832J       SHARPSBURG   NASH           14
##  4 629875C       FAYETTEVILLE CUMBERLAND     13
##  5 631426M       CHARLOTTE    MECKLENBURG    12
##  6 716184H       CHARLOTTE    MECKLENBURG     8
##  7 629881F       FAYETTEVILLE CUMBERLAND      7
##  8 726789W       CHARLOTTE    MECKLENBURG     5
##  9 631719R       BOSTIC       RUTHERFORD      4
## 10 715343J       CHARLOTTE    MECKLENBURG     4
## # i 124 more rows
```

A crossing in Rocky Mount, NC received the most complaints– 167 in a 3 year period.

### 29. Follow up question, when were all the Rocky Mount, NC complaints, was it a one time event or does it happen routinely?

```
rocky_mount_blocked <- nc_blocked_crossings %>%
  group_by(`Crossing ID`, Date, Railroad) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))

rocky_mount_blocked
```

```
## # A tibble: 271 x 4
##    `Crossing ID` Date       Railroad Count
##    <chr>         <date>     <chr>    <int>
##  1 629771V       2024-05-20 CSX         49
##  2 629771V       2024-01-03 CSX         19
##  3 629771V       2024-09-18 CSX         16
##  4 629771V       2023-09-14 CSX         11
##  5 629771V       2024-02-07 CSX          5
##  6 629767F       2023-01-15 CSX          3
##  7 629771V       2022-02-10 CSX          3
##  8 629771V       2022-09-19 CSX          3
##  9 629771V       2022-12-07 CSX          3
## 10 629771V       2022-12-30 CSX          3
## # i 261 more rows
```

```
# looks like it was dominated by a few huge delay days, all operated by CSX
```

Rocky Mount's complaints were accumulated most on 4 distinct dates: May 20, 2024 (49 complaints), January 3, 2024 (19 complaints), September 18, 2024 (16 complaints), and September 14, 2024 (11 complaints).

### 30. What percentage of North Carolina complaints involved blockages longer than 15 minutes?

```r
more_than_fifteen_minutes_nc <- nc_blocked_crossings %>%
  filter(Duration != "0-15 minutes")

percent_more_than_fifteen_nc <- nrow(more_than_fifteen_minutes_nc)/nrow(nc_blocked_crossings) * 100

percent_more_than_fifteen_nc
```

```
## [1] 87.10462
```

87% of reported complaints in NC were complaints with reported delay times of more than 15 minutes.

### 31. How many North Carolina complaints mentioned "first responders were observed being unable to cross the tracks?"

```r
nc_responders_blocked <- nc_blocked_crossings %>%
  # this string detect only works as well as it does like this bc this outputs came from a form, fixed
  filter(str_detect(`Immediate Impacts`, "FIRST RESPONDERS"))

nrow(nc_responders_blocked)
```

```
## [1] 63
```

63 complaints mentioned first responders were observed unable to cross the tracks.

### 32. What county had the most complaints that mentioned "first responders were observed being unable to cross the tracks?"

```r
nc_responders_blocked_county <- nc_responders_blocked %>%
  group_by(County) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

print(nc_responders_blocked_county)
```

```
## # A tibble: 17 x 2
##    County      Count
##    <chr>       <int>
##  1 NASH           21
##  2 MECKLENBURG    14
##  3 CUMBERLAND      8
##  4 GASTON          3
##  5 JOHNSTON        3
##  6 BRUNSWICK       2
##  7 PITT            2
##  8 ANSON           1
##  9 BEAUFORT        1
## 10 CHATHAM         1
```

```
## 11 CRAVEN          1
## 12 DURHAM          1
## 13 FORSYTH         1
## 14 HALIFAX         1
## 15 PASQUOTANK      1
## 16 ROCKINGHAM      1
## 17 RUTHERFORD      1
```

Nash County, home of Rocky Mount, had 21 reports of first responders unable to cross the tracks due to a blocked train crossing.

### 33. How many North Carolina complaints mentioned "pedestrians were observed climbing on, over, or through train cars?"

```
nc_pedestrians_blocked <- nc_blocked_crossings %>%
  filter(str_detect(`Immediate Impacts`, "PEDESTRIANS"))

nc_pedestrians_blocked
```

```
## # A tibble: 55 x 17
##    `Crossing ID` City  State Street County Railroad `Date/Time`         Duration
##    <chr>         <chr> <chr> <chr>  <chr>  <chr>    <dttm>              <chr>
##  1 470213F       WALK~ NC    OAK R~ FORSY~ NS       2022-08-30 10:02:00 1-2 hou~
##  2 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-06-25 11:48:00 16-30 m~
##  3 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-05-08 17:40:00 16-30 m~
##  4 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-04-14 12:02:00 31-60 m~
##  5 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-04-04 18:00:00 16-30 m~
##  6 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-04-04 18:00:00 31-60 m~
##  7 629771V       ROCK~ NC    TARBO~ NASH   CSX      2022-03-10 18:04:53 16-30 m~
##  8 629832J       SHAR~ NC    MAIN ~ NASH   CSX      2022-11-14 22:00:00 16-30 m~
##  9 629832J       SHAR~ NC    MAIN ~ NASH   CSX      2022-03-13 18:04:00 16-30 m~
## 10 629881F       FAYE~ NC    HAY S~ CUMBE~ CSX      2022-09-24 18:33:00 31-60 m~
## # i 45 more rows
## # i 9 more variables: Reason <chr>, `Immediate Impacts` <chr>,
## #   `Additional Comments` <chr>, Date <date>, Year <dbl>, Month <ord>,
## #   Time <chr>, Hour <int>, Peak_Commute <chr>
```

55 complaints mentioned "pedestrians were observed climbing on, over, or through train cars."

### 34. What county had the most complaints that mentioned "pedestrians were observed climbing on, over, or through train cars?"

```
nc_county_pedestrians_blocked <-nc_pedestrians_blocked %>%
  group_by(County) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

print(nc_county_pedestrians_blocked)
```

```
## # A tibble: 11 x 2
##    County      Count
##    <chr>       <int>
##  1 MECKLENBURG    25
##  2 NASH           15
##  3 CUMBERLAND      5
##  4 DURHAM          3
##  5 CHATHAM         1
##  6 CRAVEN          1
##  7 FORSYTH         1
##  8 JOHNSTON        1
##  9 PITT            1
## 10 ROCKINGHAM      1
## 11 RUTHERFORD      1
```

Of those 55 complaints, 25 were reported in Mecklenburg county and 15 were reported in Nash County.

## Novel Analysis: applying multivariable linear regression to analyze the interplay of blocked crossings with walkability scores and demographic factors of a region

### add in census data & walkability scores

```
# download the national walkability index from the epa
# used chatgpt to help me read in this folder since I've never seen data like this
# path to the .gdb folder (no trailing slash!)

gdb_path <- "data/WalkabilityIndex/Natl_WI.gdb"
st_layers(gdb_path)
```

```
## Driver: OpenFileGDB
## Available layers:
##                 layer_name geometry_type features fields
## 1 NationalWalkabilityIndex Multi Polygon   220739     29
##                                           crs_name
## 1 USA_Contiguous_Albers_Equal_Area_Conic_USGS_version
```

```
walkability <- st_read(gdb_path, layer = "NationalWalkabilityIndex")
```

```
## Reading layer 'NationalWalkabilityIndex' from data source
##   '/Users/annafetter/Desktop/plan372/final/data/WalkabilityIndex/Natl_WI.gdb'
##   using driver 'OpenFileGDB'
## Simple feature collection with 220739 features and 29 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -10434580 ymin: -83867.97 xmax: 3407868 ymax: 6755033
## Projected CRS: USA_Contiguous_Albers_Equal_Area_Conic_USGS_version
```

### preliminary mapping of national walkability scores

```
# get just walkability scores for all of the blocks
walkability_scores <- walkability %>%
  select(GEOID20, NatWalkInd) %>%
  mutate(NatWalkInd = as.numeric(NatWalkInd))
#walkability_map <- ggplot(walkability) +
  #geom_sf(aes(fill = NatWalkInd), color = NA) +
  #scale_fill_viridis_c(option = "plasma", na.value = "grey80") +
 #labs(title = "EPA National Walkability Index by Census Block Group",
       #fill = "Walkability Index") +
  #theme_minimal()

#print(walkability_map)
```

```
#the whole country is super big and super slow, here's just North Carolina walkability
nc_walk <- walkability %>%
  filter(STATEFP == "37")  # NC FIPS code

nc_walk_map <- ggplot(nc_walk) +
  geom_sf(aes(fill = NatWalkInd), color = NA) +
  scale_fill_viridis_c(option = "plasma") +
  labs(title = "Walkability Index - North Carolina",
       fill = "Index") +
  theme_minimal()

ggsave("plots/nc_walkability_map.png", plot = nc_walk_map, width = 10, height = 6)
```

### now actually get the locations of the train crossings by ID

```
# https://data.transportation.gov/Railroads/Crossing-Inventory-Data-Form-71-Current/m2f8-22s6/explore

locations_crossing_IDS <- read_csv("data/crossing_inventory_may_2025.csv")
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)

## Rows: 437935 Columns: 20
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (13): Crossing ID, State Code, County Name, City Code, City Name, City D...
## dbl  (7): Block Number, Crossing Purpose Code, Crossing Type Code, Crossing ...
##
## 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.
```

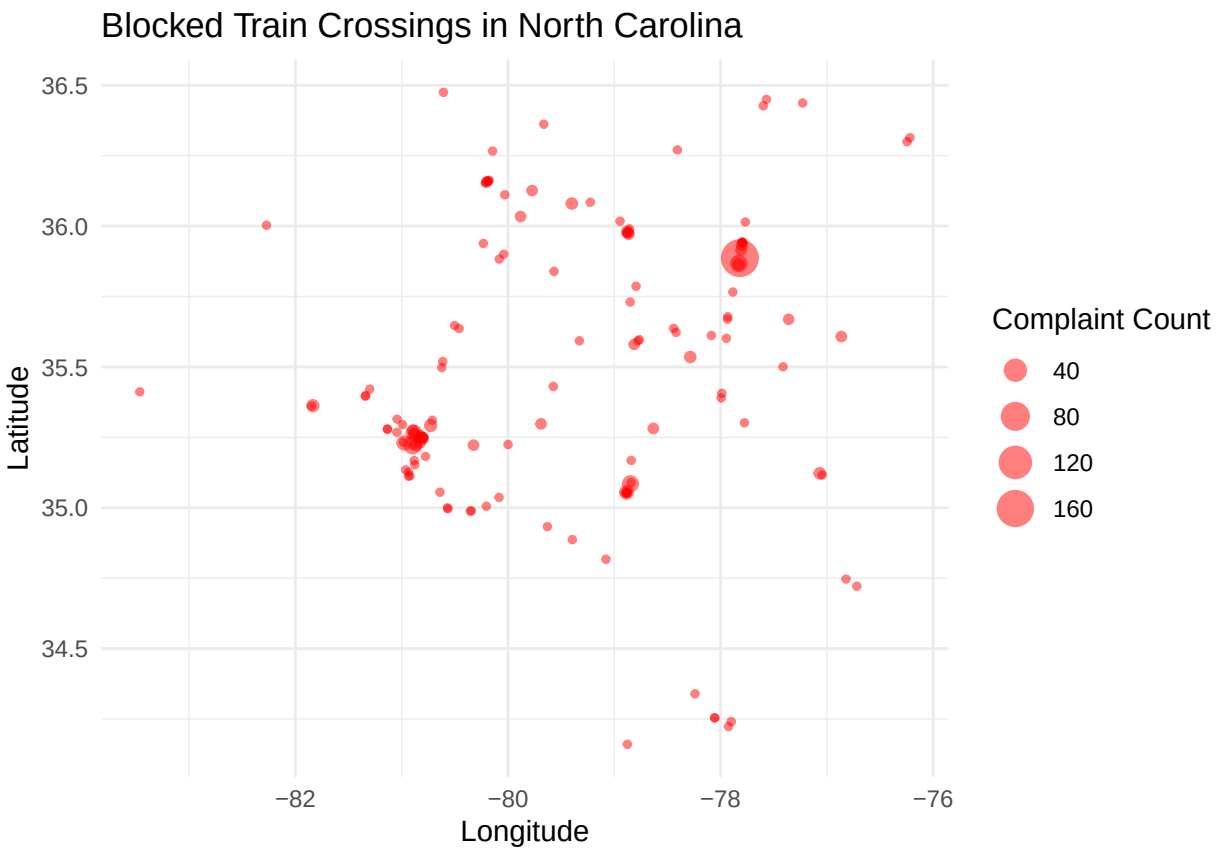
```r
# now that I've read in the data I've narrowed down the columns I want
locations_crossing_IDS <- locations_crossing_IDS %>%
  select(`Crossing ID`,`Crossing Type`, `Crossing Purpose`, Longitude, Latitude)

# now we're going to join the blocked crossing dataset to locations_crossing_ID by ID, we need to match

blocked_crossings_location <- blocked_crossings %>%
  left_join(locations_crossing_IDS, by = c("Crossing ID" = "Crossing ID"))
```

```
## Warning in left_join(., locations_crossing_IDS, by = c(‘Crossing ID‘ = "Crossing ID")): Detected an u
## i Row 33722 of ‘x‘ matches multiple rows in ‘y‘.
## i Row 296931 of ‘y‘ matches multiple rows in ‘x‘.
## i If a many-to-many relationship is expected, set ‘relationship =
##   "many-to-many"‘ to silence this warning.
```

```r
# for mapping purposes let's make a dataframe only of crossings that have complaints
crossings_with_complaints <- blocked_crossings_location %>%
  group_by(`Crossing ID`, Longitude, Latitude, `Crossing Type`, `Crossing Purpose`, State) %>%
  summarise(complaint_count = n(), .groups = "drop") %>%
  filter(!is.na(Longitude) & !is.na(Latitude))
```

### now let's map all of our blocked crossings

```r
# let's do this just in NC for mapping ease
crossings_with_complaints_nc <- crossings_with_complaints %>%
  filter(State == "NC")

nc_crossings_map <- ggplot() +
  geom_point(data = crossings_with_complaints_nc, aes(x = Longitude, y = Latitude, size = complaint_cou
  scale_fill_viridis_c(option = "plasma") +
  labs(title = "Blocked Train Crossings in North Carolina",
       size = "Complaint Count") +
  theme_minimal()

print(nc_crossings_map)
```




```
st_crs(nc_walk)
```

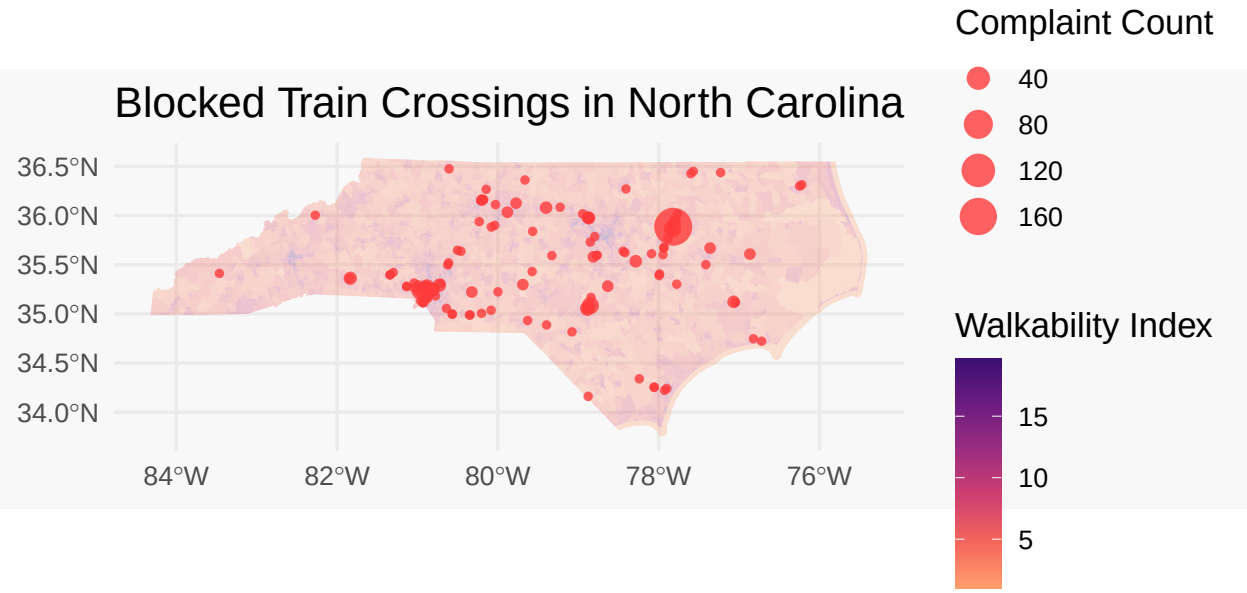
```
## Coordinate Reference System:
##   User input: USA_Contiguous_Albers_Equal_Area_Conic_USGS_version
##   wkt:
## PROJCRS["USA_Contiguous_Albers_Equal_Area_Conic_USGS_version",
##     BASEGEOGCRS["NAD83",
##         DATUM["North American Datum 1983",
##             ELLIPSOID["GRS 1980",6378137,298.257222101,
##                 LENGTHUNIT["metre",1]]],
##         PRIMEM["Greenwich",0,
##             ANGLEUNIT["degree",0.0174532925199433]],
##         ID["EPSG",4269]],
##     CONVERSION["USA_Contiguous_Albers_Equal_Area_Conic_USGS_version",
##         METHOD["Albers Equal Area",
##             ID["EPSG",9822]],
##         PARAMETER["Latitude of false origin",23,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8821]],
##         PARAMETER["Longitude of false origin",-96,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8822]],
##         PARAMETER["Latitude of 1st standard parallel",29.5,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8823]],
```

```
##         PARAMETER["Latitude of 2nd standard parallel",45.5,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8824]],
##         PARAMETER["Easting at false origin",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8826]],
##         PARAMETER["Northing at false origin",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8827]]],
##     CS[Cartesian,2],
##         AXIS["(E)",east,
##             ORDER[1],
##             LENGTHUNIT["metre",1]],
##         AXIS["(N)",north,
##             ORDER[2],
##             LENGTHUNIT["metre",1]],
##     USAGE[
##         SCOPE["Not known."],
##         AREA["United States (USA) - CONUS onshore - Alabama; Arizona; Arkansas; California; Colorado
##         BBOX[24.41,-124.79,49.38,-66.91]],
##     ID["ESRI",102039]]
```

```
st_crs(crossings_with_complaints_nc)
```

```
## Coordinate Reference System: NA
```

```
nc_walk <- st_transform(nc_walk, crs = 4326)
crossings_nc_sf <- st_as_sf(crossings_with_complaints_nc,
                            coords = c("Longitude", "Latitude"),
                            crs = 4326)

#asked chatgpt for some styling help, focusing on making walkability less visually dominant
nc_crossings_map <- ggplot() +
  geom_sf(data = nc_walk, aes(fill = NatWalkInd), color = NA, alpha = 0.3) +
  # emphasize blocked crossings
  geom_sf(data = crossings_nc_sf,
          aes(size = complaint_count),
          color = "#FF3B3B",  # bold red
          alpha = 0.8) +
  # lighter, lower-contrast palette for walkability
  scale_fill_viridis_c(option = "magma", direction = -1, begin = 0.2, end = 0.8) +
  labs(title = "Blocked Train Crossings in North Carolina",
       fill = "Walkability Index",
       size = "Complaint Count") +
  theme_minimal(base_size = 13) +
  theme(
    panel.background = element_rect(fill = "#f8f8f8", color = NA),
    plot.background = element_rect(fill = "#f8f8f8", color = NA),
    legend.key = element_blank()
  )

print(nc_crossings_map)
```




```
ggsave("plots/nc_crossings_map.png", plot = nc_crossings_map, width = 10, height = 6)
```

### now let's map with census blocks

```
# load in census data
# because block data is so intensive, we're going to use the top 5 states based off number of complaint

top_5_states_blocked <- states_most_blocked %>%
  slice_head(n = 5) %>%
  pull(State)

# used chatgpt to get me a function so I could run this on all the states if I wanted to, starting with
# Function to get total population at the block group level
get_bg_population <- function(state_abbr) {
  get_acs(
    geography = "block group",
    variables = c(population = "B01003_001"),  # total population
    state = state_abbr,
    year = 2022,
    survey = "acs5",
    geometry = FALSE
  )
}
```

```r
# Pull and reshape
acs_top_5_states <- map_df(top_5_states_blocked, get_bg_population) %>%
  select(GEOID, variable, estimate) %>%
  pivot_wider(names_from = variable, values_from = estimate)
```

```
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
```

```r
# I ran into the problem where you cant access car access at the local level. I've decided to mix block
#"Because the U.S. Census Bureau does not consistently report household vehicle access at the block gro
# Function to pull tract-level data

get_tract_acs <- function(state_abbr) {
  get_acs(
    geography = "tract",
    variables = c(
      population = "B01003_001",
      median_income = "B19013_001",
      no_vehicle = "B08201_002",
      total_households = "B08201_001"
    ),
    state = state_abbr,
    year = 2022,
    survey = "acs5",
    geometry = FALSE
  )
}

# Download and pivot
acs_top5_tracts <- map_df(top_5_states_blocked, get_tract_acs) %>%
  select(GEOID, variable, estimate) %>%
  pivot_wider(names_from = variable, values_from = estimate) %>%
  mutate(
    pct_no_vehicle = no_vehicle / total_households * 100
  )
```

```
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
## Getting data from the 2018-2022 5-year ACS
```

### now we're going to combine the data to build the model

```r
# the regression is going to use geographic data from the 5 states with the most complaints

# first join census data with walkability score, need this to map the blocks correctly
# Filter walkability to just the top 5 states, r kept crashing so I had to limit my scope
```

```r
fips_top5 <- fips_codes %>%
  filter(state %in% top_5_states_blocked) %>%
  distinct(state, state_code) %>%
  pull(state_code)

walkability_scores_top5 <- walkability_scores %>%
  filter(substr(GEOID20, 1, 2) %in% fips_top5)

# Now do the join
walkability_acs <- walkability_scores_top5 %>%
  select(GEOID20, NatWalkInd, Shape) %>%
  left_join(acs_top_5_states, by = c("GEOID20" = "GEOID"))  # population only

# Calculate population density, add to walkability_acs
walkability_acs <- walkability_acs %>%
  mutate(area_km2 = st_area(.) %>% units::set_units("km^2") %>% as.numeric()) %>%
  mutate(population_density = population / area_km2)

# add tract geoid to walkability for when I join it on tracts later
walkability_acs <- walkability_acs %>%
  mutate(tract_geoid = substr(GEOID20, 1, 11))

# join in tract-level ACS demographic data
walkability_acs <- walkability_acs %>%
  left_join(acs_top5_tracts, by = c("tract_geoid" = "GEOID"))

# Convert to sf
crossings_sf <- blocked_crossings_location %>%
  filter(!is.na(Longitude), !is.na(Latitude)) %>%
  st_as_sf(coords = c("Longitude", "Latitude"), crs = 4326)

# Make sure walkability_acs has geometry - if not, add it back from original shapefile
walkability_geo <- st_as_sf(walkability_acs, sf_column_name = "Shape", crs = 4326)
walkability_geo <- st_transform(walkability_geo, crs = 4326)

# Spatial join: assign GEOID to each crossing, carry over needed columns
crossings_census_walk <- st_join(
  crossings_sf,
  walkability_geo[, c("GEOID20", "NatWalkInd", "population_density", "median_income", "pct_no_vehicle")]
)

crossings_census_walk_5 <- crossings_census_walk %>%
  filter(substr(GEOID20, 1, 2) %in% fips_top5)
```

### let's map these states individually

```r
# mapping these states
# create a function to plot each state
plot_state_crossings <- function(state_abbr) {
  # Filter state-level crossing data
  state_data <- crossings_census_walk_5 %>%
```

```
    filter(State == state_abbr)

  # Get state FIPS code
  state_fips <- fips_codes %>%
    filter(state == state_abbr) %>%
    distinct(state_code) %>%
    pull(state_code)

  # Filter walkability data to this state only
  walkability_state <- walkability_geo %>%
    filter(substr(GEOID20, 1, 2) == state_fips)

  # Crop to the extent of the state for framing
  state_bbox <- st_bbox(walkability_state)

  # Build the map
  state_map <- ggplot() +
    geom_sf(data = walkability_state, aes(fill = NatWalkInd), color = NA, alpha = 0.3) +
    geom_sf(data = state_data, color = "#FF3B3B", alpha = 0.8) +
    scale_fill_viridis_c(option = "magma", direction = -1, begin = 0.2, end = 0.8) +
    coord_sf(xlim = c(state_bbox$xmin, state_bbox$xmax),
             ylim = c(state_bbox$ymin, state_bbox$ymax),
             expand = FALSE) +
    labs(title = paste("Blocked Train Crossings in", state_abbr),
         fill = "Walkability Index") +
    theme_minimal(base_size = 13) +
    theme(
      panel.background = element_rect(fill = "#f8f8f8", color = NA),
      plot.background = element_rect(fill = "#f8f8f8", color = NA),
      legend.key = element_blank()
    )

  ggsave(paste0("plots/", state_abbr, "_crossings_map.png"), plot = state_map, width = 10, height = 6)
}

# loop through the top 5 states and save each plot
for (state in top_5_states_blocked) {
  plot_state_crossings(state)
}
```

**Now let's finally get to the regression. I will be aggregating into crossing level analysis.**

```
# use the new crossing census walk 5 dataframe

# set long delay as those longer than 15 minutes
crossings_census_walk_5 <- crossings_census_walk_5 %>%
  mutate(long_delay = Duration != "0-15 minutes") %>%
  filter(`Crossing Type` != "Private") %>% # remove private crossings, I noticed many of these also did
  mutate(`Immediate Impacts` =replace_na(`Immediate Impacts`, "")) # this is so rows with no immediate
```

```r
# I need to fix my regression data, this only looks at crossings with more than 5 complaints
crossing_summary_5_states <- crossings_census_walk_5 %>%
  st_drop_geometry() %>%
  group_by(`Crossing ID`) %>%
  summarise(
    total_complaints = n(),
    climb_pct = mean(str_detect(`Immediate Impacts`, regex("climb", ignore_case = TRUE)), na.rm = TRUE)
    long_delay_pct = mean(long_delay, na.rm = TRUE),
    walkability = first(NatWalkInd),
    pop_density = first(population_density),
    median_income = first(median_income),
    pct_no_vehicle = first(pct_no_vehicle),
    crossing_purpose = first(`Crossing Purpose`)
  ) %>%
  filter(total_complaints >= 5)
```

### building the model

```r
model <- lm(climb_pct ~ walkability + pop_density + median_income + pct_no_vehicle + long_delay_pct,
            data = crossing_summary_5_states)

summary(model)
```

```
##
## Call:
## lm(formula = climb_pct ~ walkability + pop_density + median_income +
##     pct_no_vehicle + long_delay_pct, data = crossing_summary_5_states)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46876 -0.13978 -0.03875  0.09732  0.67400
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.356e-01  5.517e-02  -2.457 0.014190 *
## walkability     1.512e-02  2.054e-03   7.357 4.43e-13 ***
## pop_density     2.367e-05  7.631e-06   3.102 0.001984 **
## median_income  -9.694e-07  2.916e-07  -3.324 0.000926 ***
## pct_no_vehicle  1.629e-03  1.155e-03   1.410 0.158930
## long_delay_pct  2.772e-01  4.792e-02   5.784 1.02e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.201 on 849 degrees of freedom
##   (246 observations deleted due to missingness)
## Multiple R-squared:  0.2183, Adjusted R-squared:  0.2137
## F-statistic: 47.42 on 5 and 849 DF,  p-value: < 2.2e-16
```

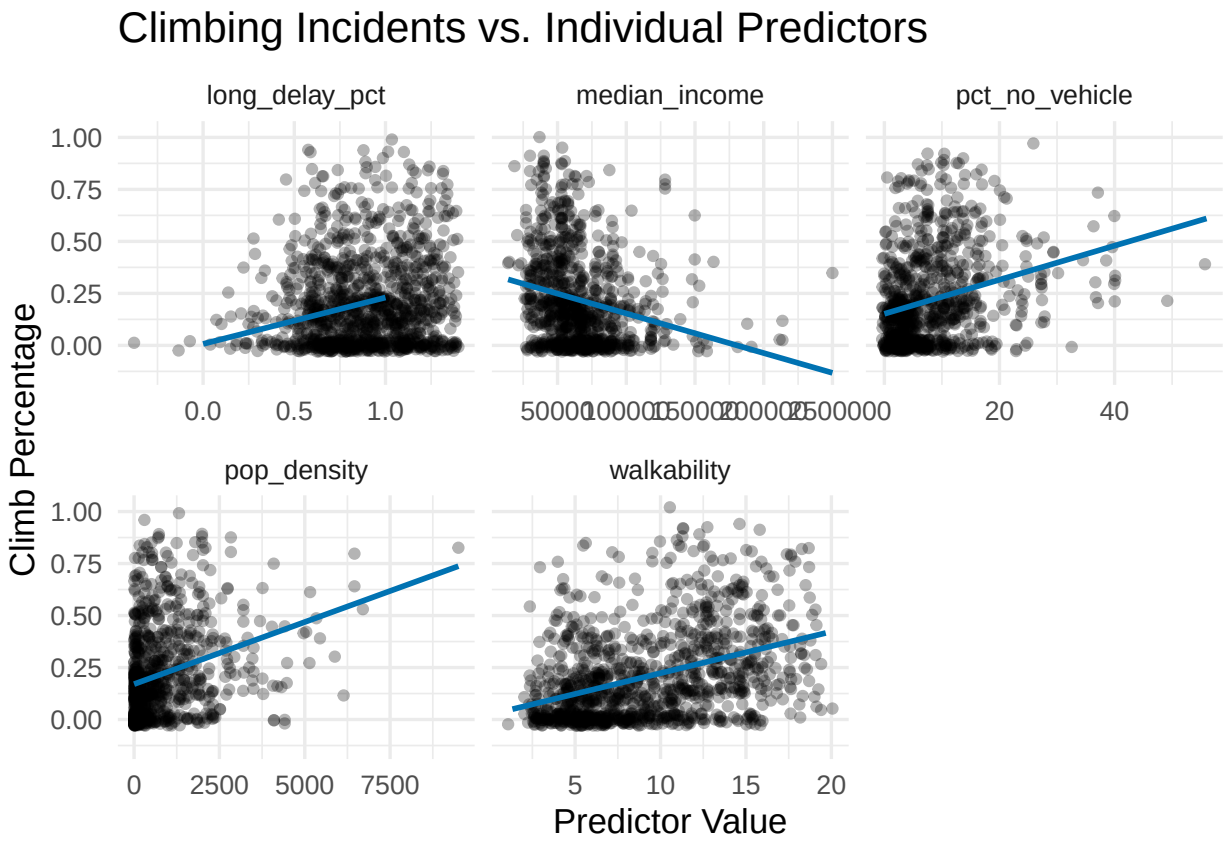
### more plotting

```r
# Reshape your data: long format for predictors
plot_data <- crossing_summary_5_states %>%
  select(climb_pct, walkability, pop_density, median_income, pct_no_vehicle, long_delay_pct) %>%
  pivot_longer(cols = -climb_pct, names_to = "predictor", values_to = "value")

# Plot with facets
ggplot(plot_data, aes(x = value, y = climb_pct)) +
  geom_jitter(alpha = 0.3, width = 0.4, height = 0.03, color = "black") +
  geom_smooth(method = "lm", se = FALSE, color = "#0072B2") +
  facet_wrap(~ predictor, scales = "free_x") +
  labs(
    title = "Climbing Incidents vs. Individual Predictors",
    x = "Predictor Value",
    y = "Climb Percentage"
  ) +
  theme_minimal(base_size = 13)
```

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

```
## Warning: Removed 708 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 708 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




### Exploratory plots, for presentation

```
library(patchwork) # for combining plots
```

```
## Warning: package 'patchwork' was built under R version 4.3.3
```

```
# Plot 1: Walkability
p1 <- ggplot(crossing_summary_5_states, aes(x = walkability, y = climb_pct)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = FALSE, color = "#0072B2") +
  labs(title = "Walkability vs. Climb %", x = "Walkability Score", y = "Climb Percentage") +
  theme_minimal()

# Plot 2: Median Income
p2 <- ggplot(crossing_summary_5_states, aes(x = median_income, y = climb_pct)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = FALSE, color = "#0072B2") +
  labs(title = "Median Income vs. Climb %", x = "Median Income ($)", y = NULL) +
  theme_minimal()

# Plot 3: Long Delay %
p3 <- ggplot(crossing_summary_5_states, aes(x = long_delay_pct, y = climb_pct)) +
  geom_point(alpha = 0.4) +
```

```r
  geom_smooth(method = "lm", se = FALSE, color = "#0072B2") +
  labs(title = "Long Delay % vs. Climb %", x = "Long Delay Proportion", y = "Climb Percentage") +
  theme_minimal()

# Plot 4: Population Density (optional log-scale)
p4 <- ggplot(crossing_summary_5_states, aes(x = pop_density, y = climb_pct)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm", se = FALSE, color = "#0072B2") +
  labs(title = "Population Density vs. Climb %", x = "People per km²", y = NULL) +
  theme_minimal()

# Combine using patchwork
(p1 | p2) / (p3 | p4)
```

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

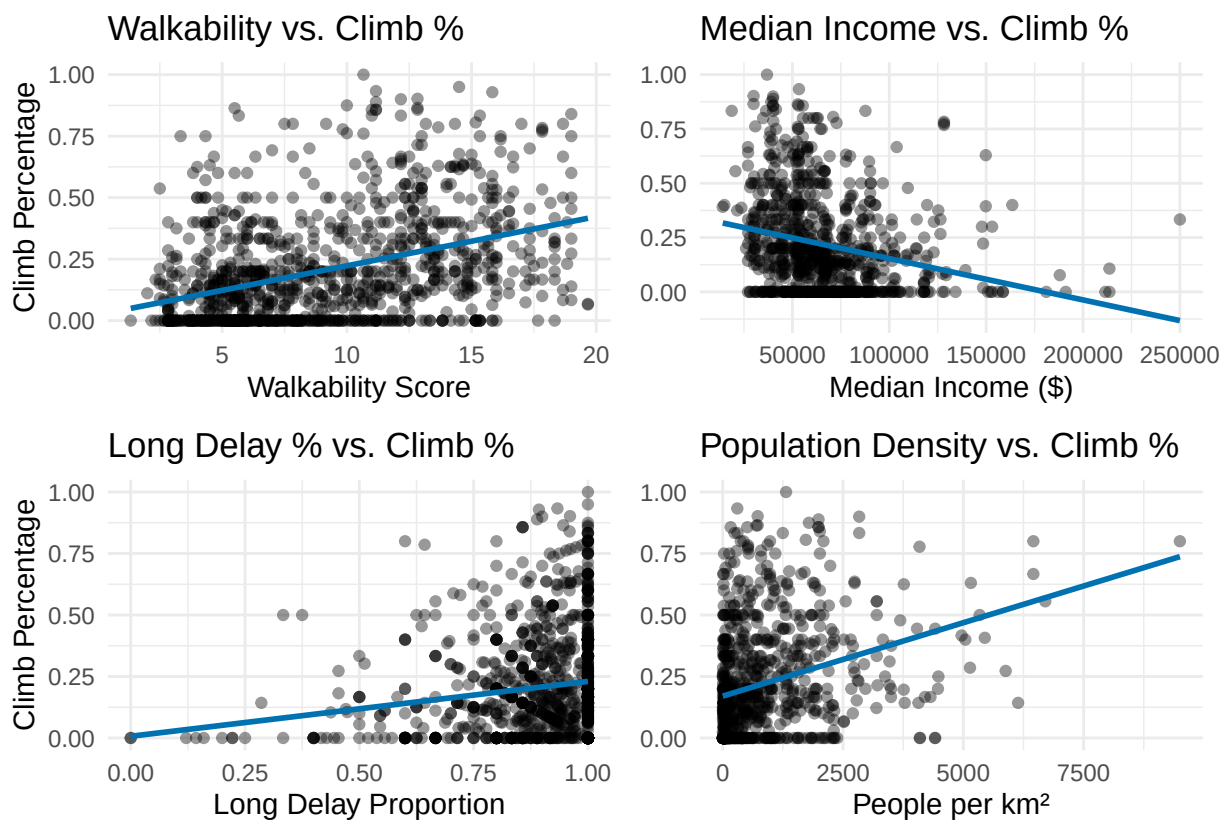
```
## Warning: Removed 231 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 231 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

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

```
## Warning: Removed 246 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 246 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
ggsave("plots/exploratory_plots.png", width = 12, height = 8)
```

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

## Warning: Removed 231 rows containing non-finite outside the scale range
## (`stat_smooth()`).

## Warning: Removed 231 rows containing missing values or values outside the scale range
## (`geom_point()`).

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

## Warning: Removed 246 rows containing non-finite outside the scale range
## (`stat_smooth()`).

## Warning: Removed 246 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

## for real world tie in, find top risk crossings

```r
high_risk_crossings <- crossing_summary_5_states %>%
  filter(total_complaints >= 5) %>%
  mutate(
    risk_score = (
      scale(climb_pct)[, 1] * 1.5 +               # Main outcome of interest
      scale(walkability)[, 1] * 1 +
      scale(long_delay_pct)[, 1] * 1 +
      scale(total_complaints)[, 1] * 1.2 -         # Elevate frequent complaints
      scale(median_income)[, 1] * 0.5              # Lower-income vulnerablity
    )
  ) %>%
  arrange(desc(risk_score)) %>%
  slice_head(n = 20)


high_risk_crossings <- high_risk_crossings %>%
  left_join(
    blocked_crossings_location %>%
      select(`Crossing ID`, City, State, County, Street, Longitude, Latitude),
    by = "Crossing ID"
  ) %>%
  distinct(`Crossing ID`, .keep_all = TRUE)

print(high_risk_crossings)
```

```
## # A tibble: 20 x 16
##    `Crossing ID` total_complaints climb_pct long_delay_pct walkability
##    <chr>                    <int>     <dbl>          <dbl>       <dbl>
##  1 859522Y                   1388     0.266          0.818        14.7
##  2 288226J                    928     0.630          0.962        15.2
##  3 288224V                    861     0.686          0.935        16.2
##  4 288221A                    783     0.737          0.925        16.2
##  5 859523F                    669     0.629          0.954        14.7
##  6 288227R                    517     0.783          0.986        17.8
##  7 869223U                    572     0.531          0.920        15.8
##  8 869221F                    695     0.455          0.885        11.3
##  9 859524M                    504     0.478          0.944        15.8
## 10 859521S                    346     0.636          0.974        14.7
## 11 859518J                    298     0.678          0.980        14.5
## 12 755640L                    346     0.624          0.931        14.3
## 13 288228X                    289     0.768          0.962        17.8
## 14 859527H                    264     0.674          0.951        15
## 15 288229E                    232     0.776          0.974        17.8
## 16 448676Y                    196     0.760          1            10.7
## 17 859529W                     25     0.84           0.96         19
## 18 732190C                     51     0.902          0.961        12.8
## 19 299611T                      6     0.833          1            15.3
## 20 663400Y                      8     1              1            10.7
## # i 11 more variables: pop_density <dbl>, median_income <dbl>,
## #   pct_no_vehicle <dbl>, crossing_purpose <chr>, risk_score <dbl>, City <chr>,
## #   State <chr>, County <chr>, Street <chr>, Longitude <dbl>, Latitude <dbl>
```

```r
top_5_crossings <- high_risk_crossings %>%
  slice_max(order_by = risk_score, n = 5)

top_5_crossings_sf <- top_5_crossings %>%
  filter(!is.na(Longitude), !is.na(Latitude)) %>%
  st_as_sf(coords = c("Longitude", "Latitude"), crs = 4326)

# interesting, all 5 of the top risk crossings are in Houston
```

## map high risk crossings in Houston, with roads & walkability scores

```r
# Get bounding box with some padding
houston_bbox <- st_bbox(st_union(top_5_crossings_sf)) %>%
  as.numeric() %>%
  setNames(c("xmin", "ymin", "xmax", "ymax"))

# 1 km buffer around all top 5 crossings
houston_area <- st_union(top_5_crossings_sf) %>%
  st_buffer(dist = 1000)

# Filter walkability data to just that area
houston_walkability <- walkability_geo[houston_area, ]

# Expand it a bit for breathing room
buffer_deg <- 0.02
houston_bbox_expanded <- c(
  left   = houston_bbox["xmin"] - buffer_deg,
  bottom = houston_bbox["ymin"] - buffer_deg,
  right  = houston_bbox["xmax"] + buffer_deg,
  top    = houston_bbox["ymax"] + buffer_deg
)

houston_streets <- opq(bbox = houston_bbox_expanded) %>%
  add_osm_feature(key = "highway") %>%
  osmdata_sf()

roads_sf <- houston_streets$osm_lines

ggplot() +
  # Walkability fill
  geom_sf(data = houston_walkability, aes(fill = NatWalkInd), alpha = 0.4, color = NA) +
  # Roads
  geom_sf(data = roads_sf, color = "gray60", size = 0.2) +
  # Top crossings
  geom_sf(data = top_5_crossings_sf, color = "red", size = 3) +
  geom_sf_text(data = top_5_crossings_sf, aes(label = `Crossing ID`), nudge_y = 0.0015, size = 3) +
  scale_fill_viridis_c(option = "magma") +
  labs(
    title = "High-Risk Train Crossings in Houston, TX",
    fill = "Walkability Index"
  ) +
```

```
  coord_sf(xlim = c(houston_bbox_expanded["left"], houston_bbox_expanded["right"]),
           ylim = c(houston_bbox_expanded["bottom"], houston_bbox_expanded["top"])) +
  theme_minimal()
```

```
## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data
```