PBL Group 2: CTA Rail Station Crime Exposure & Ridership Analysis

SOSC 13220: SSI Spatial Analysis II — Winter 2026

Andy Pincus, Blair Peng, Julien Lee

2026-02-25

Contents

Overview of Steps

This Rmd follows these explicit steps:

1. **Load packages**
2. **Download and load data** (CTA L stops, ridership, crime, community areas, police stations)
3. **Clean and prepare data** (filter, aggregate, spatialize)
4. **Create buffer zones** around CTA rail stations (quarter-mile and half-mile)
5. **Spatial join**: Count crimes within each buffer
6. **Merge** crime counts with ridership data
7. **Non-spatial EDA**: Scatter plots, correlation, summary stats
8. **LISA / ESDA**: Spatial clustering of crime by census tract (per feedback)
9. **Buffer map visualization**: Crime exposure near stations
10. **Priority intervention map**: High-crime / low-ridership stations
11. **Regression**: Correlation between ridership and crime count (supplementary)

Step 1: Load Packages

```
# Install if needed (uncomment):
# install.packages(c("sf", "dplyr", "ggplot2", "tmap", "spdep", "jsonlite", "tidyr", "viridis"))

library(sf)
library(dplyr)
library(ggplot2)
library(tmap)
library(spdep)
library(jsonlite)
library(tidyr)
library(viridis)

cat("All packages loaded.\n")
```

```
## All packages loaded.
```

Step 2: Download and Load Data

We pull everything directly from the Chicago Data Portal APIs so the analysis is reproducible.

2a. CTA L Station Locations (with lat/lon)

```
# Dataset: CTA System Information - List of 'L' Stops
# Portal ID: 8pix-ypme
# This has lat/lon in a "Location" column plus line info

cta_stops_raw <- read.csv(
  "https://data.cityofchicago.org/api/views/8pix-ypme/rows.csv?accessType=DOWNLOAD",
  stringsAsFactors = FALSE
)

cat("CTA L stops loaded:", nrow(cta_stops_raw), "rows\n")
```

```
## CTA L stops loaded: 302 rows
```

```
head(cta_stops_raw)
```

```
##   STOP_ID DIRECTION_ID                      STOP_NAME STATION_NAME
## 1   30263            E            Oak Park (63rd-bound)     Oak Park
## 2   30152            W Central Park (54th/Cermak-bound) Central Park
## 3   30148            S           Granville (95th-bound)    Granville
## 4   30044            N        Cumberland (O'Hare-bound)   Cumberland
## 5   30007            N        Quincy/Wells (Inner Loop) Quincy/Wells
## 6   30172            S         O'Hare (Forest Pk-bound)       O'Hare
##                           STATION_DESCRIPTIVE_NAME MAP_ID   ADA   RED  BLUE
## 1                           Oak Park (Green Line)  41350 false false false
## 2                         Central Park (Pink Line)  40780  true false false
## 3                             Granville (Red Line)  40760  true  true false
## 4                          Cumberland (Blue Line)  40230  true false  true
## 5 Quincy/Wells (Brown, Orange, Purple & Pink lines)  40040  true false false
## 6                               O'Hare (Blue Line)  40890  true false  true
##       G   BRN     P     Y   Pnk     O                    Location
## 1  true false false false false false     (41.886988, -87.793783)
## 2 false false false false  true false     (41.853839, -87.714842)
## 3 false false false false false false     (41.993664, -87.659202)
## 4 false false false false false false     (41.984246, -87.838028)
## 5 false false  true false  true  true      (41.878723, -87.63374)
## 6 false false false false false false (41.97766526, -87.90422307)
```

Parse coordinates and aggregate to station level

The L stops dataset has multiple rows per station (one per direction/platform). We need to aggregate to unique stations using `MAP_ID` (the station identifier).

```r
# Parse lat/lon from the Location column "(lat, lon)"
cta_stops_raw <- cta_stops_raw %>%
  filter(!is.na(Location) & Location != "") %>%
  mutate(
    lat = as.numeric(gsub("\\(([^,]+),.*", "\\1", Location)),
    lon = as.numeric(gsub(".*,\\s*([^)]+)\\)", "\\1", Location))
  )

# Aggregate to unique stations (one point per station)
# MAP_ID is the station-level identifier; STATION_NAME is the station name
cta_stations <- cta_stops_raw %>%
  group_by(MAP_ID, STATION_NAME) %>%
  summarise(
    lat = mean(lat, na.rm = TRUE),
    lon = mean(lon, na.rm = TRUE),
    # Track which lines serve this station
    has_red    = any(RED == TRUE),
    has_blue   = any(BLUE == TRUE),
    has_green  = any(G == TRUE),
    has_brown  = any(BRN == TRUE),
    has_purple = any(P == TRUE),
    has_yellow = any(Y == TRUE),
    has_pink   = any(Pnk == TRUE),
    has_orange = any(O == TRUE),
    .groups = "drop"
  )

cat("Unique CTA rail stations:", nrow(cta_stations), "\n")
```

```
## Unique CTA rail stations: 144
```

```r
# Convert to sf object (CRS 4326 = WGS84)
cta_stations_sf <- st_as_sf(cta_stations, coords = c("lon", "lat"), crs = 4326)

# Transform to Illinois State Plane East (feet) for accurate distance buffering
cta_stations_proj <- st_transform(cta_stations_sf, crs = 3435)
```

2b. CTA L Station Ridership (Annual Totals)

```r
# Dataset: CTA - Ridership - 'L' Station Entries - Daily Totals
# Portal ID: 5neh-572f
# WARNING: This dataset is VERY large (millions of rows).
# We use the Socrata API to filter to a single year.
#
# Alternative: Download from the portal pre-filtered, or use the
# monthly day-type averages dataset (t2rn-p8d7) which is smaller.

# OPTION A: Use the Socrata API with SoQL to filter to 2023
# (adjust year as needed)
ridership_url <- "https://data.cityofchicago.org/resource/5neh-572f.csv?$where=date%20between%20%272023-
```

```r
ridership_raw <- read.csv(ridership_url, stringsAsFactors = FALSE)
cat("Ridership rows loaded:", nrow(ridership_raw), "\n")
```

```
## Ridership rows loaded: 52195
```

```r
# Aggregate to annual total rides per station
ridership_annual <- ridership_raw %>%
  group_by(station_id, stationname) %>%
  summarise(
    total_rides = sum(rides, na.rm = TRUE),
    avg_daily_rides = mean(rides, na.rm = TRUE),
    .groups = "drop"
  )

cat("Stations with ridership data:", nrow(ridership_annual), "\n")
```

```
## Stations with ridership data: 143
```

```r
head(ridership_annual %>% arrange(desc(total_rides)))
```

```
## # A tibble: 6 x 4
##   station_id stationname    total_rides avg_daily_rides
##        <int> <chr>                <int>           <dbl>
## 1      41660 Lake/State         3101354           8497.
## 2      40890 O'Hare Airport     2780200           7617.
## 3      40380 Clark/Lake         2498459           6845.
## 4      41450 Chicago/State      2375146           6507.
## 5      41220 Fullerton          2358244           6461.
## 6      40260 State/Lake         2345432           6426.
```

Join ridership to station locations

```r
# The key to join is MAP_ID (in cta_stations) = station_id (in ridership)
cta_stations_proj <- cta_stations_proj %>%
  left_join(ridership_annual, by = c("MAP_ID" = "station_id"))

cat("Stations with ridership joined:", sum(!is.na(cta_stations_proj$total_rides)),
    "out of", nrow(cta_stations_proj), "\n")
```

```
## Stations with ridership joined: 143 out of 144
```

```r
# Check for any unmatched stations
unmatched <- cta_stations_proj %>% filter(is.na(total_rides))
if (nrow(unmatched) > 0) {
  cat("Unmatched stations:\n")
  print(unmatched$STATION_NAME)
}
```

```
## Unmatched stations:
## [1] "Damen"
```

2c. Chicago Crime Data (2023)

```
# Dataset: Crimes - 2001 to Present
# Portal ID: ijzp-q8t2
# Use SoQL to filter to 2023 and only get needed columns
# Adjust the year to match your ridership year

crime_url <- "https://data.cityofchicago.org/resource/ijzp-q8t2.csv?$where=year%3D2023%20AND%20latitude
```

```
crime_raw <- read.csv(crime_url, stringsAsFactors = FALSE)
cat("Crime records loaded:", nrow(crime_raw), "\n")
```

```
## Crime records loaded: 261243
```

```
# If you hit the 500k limit, you may need to paginate or download manually
# Check: are we at exactly 500k?
if (nrow(crime_raw) == 500000) {
  cat("WARNING: Hit API limit. Consider downloading the full filtered CSV manually.\n")
}

# Convert to sf
crime_sf <- st_as_sf(crime_raw,
                     coords = c("longitude", "latitude"),
                     crs = 4326)

# Project to match stations
crime_proj <- st_transform(crime_sf, crs = 3435)

cat("Crime points spatialized:", nrow(crime_proj), "\n")
```

```
## Crime points spatialized: 261243
```

2d. Community Areas (for basemap + LISA analysis)

```
# Dataset: Boundaries - Community Areas (current)
# Portal ID: cauq-8yn6
list.files("~/Downloads", pattern = "oundaries|community|cauq")
```

```
## character(0)
```

```
comm_areas <- st_read("comm_area.geojson", quiet = TRUE)

comm_areas_proj <- st_transform(comm_areas, crs = 3435)
cat("Community areas loaded:", nrow(comm_areas_proj), "\n")
```

```
## Community areas loaded: 77
```

2e. Police Station Locations (optional context layer)

```
# Dataset: Police Stations
# Portal ID: z8bn-74gv
police_url <- "https://data.cityofchicago.org/resource/z8bn-74gv.csv?$limit=50"
police_raw <- read.csv(police_url, stringsAsFactors = FALSE)

# Check column names - may have latitude/longitude or location
cat("Police station columns:", paste(names(police_raw), collapse = ", "), "\n")
```

```
## Police station columns: district, district_name, address, city, state, zip, website, phone, fax, tty
```

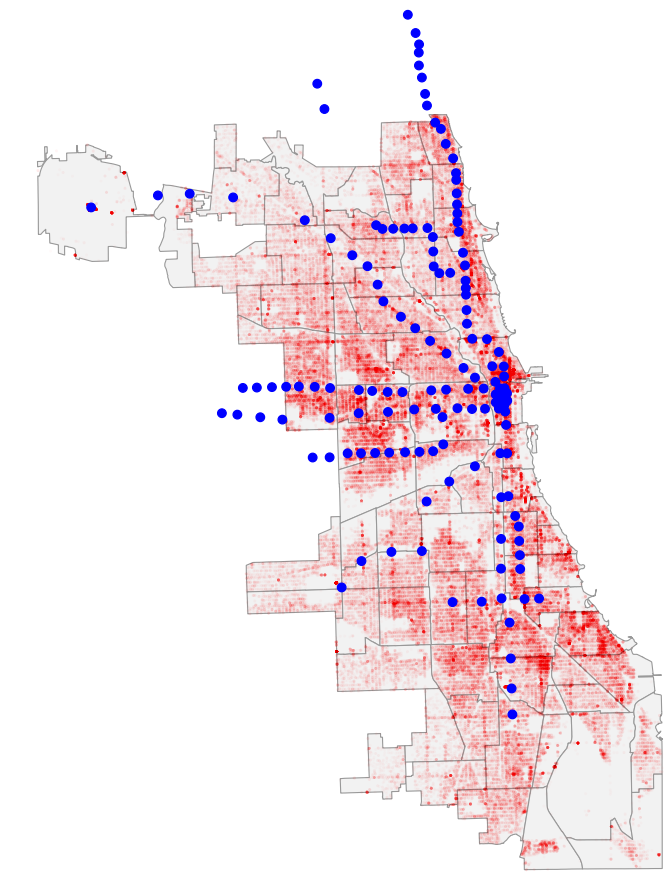
```
# Spatialize (adjust column names if needed)
if ("latitude" %in% names(police_raw) & "longitude" %in% names(police_raw)) {
  police_sf <- police_raw %>%
    filter(!is.na(latitude) & !is.na(longitude)) %>%
    st_as_sf(coords = c("longitude", "latitude"), crs = 4326)
  police_proj <- st_transform(police_sf, crs = 3435)
  cat("Police stations loaded:", nrow(police_proj), "\n")
} else {
  cat("NOTE: Check police station data columns for lat/lon fields.\n")
  print(head(police_raw))
}
```

```
## Police stations loaded: 23
```

Step 3: Verify Data Alignment

Quick plot to confirm everything overlaps correctly.

```
ggplot() +
  geom_sf(data = comm_areas_proj, fill = "grey95", color = "grey60", linewidth = 0.3) +
  geom_sf(data = crime_proj, color = "red", alpha = 0.02, size = 0.1) +
  geom_sf(data = cta_stations_proj, color = "blue", size = 2) +
  labs(title = "Data Verification: Community Areas, Crime Points, and CTA Rail Stations",
       subtitle = "CRS: EPSG 3435 (Illinois State Plane East)") +
  theme_void()
```



Step 4: Create Buffer Zones Around CTA Rail Stations

We test two buffer distances:

- **Quarter-mile (1,320 feet)**: Immediate station vicinity
- **Half-mile (2,640 feet)**: Standard transit walkability catchment

```
# Quarter-mile buffer (in feet, since CRS 3435 uses feet)
buffer_quarter <- st_buffer(cta_stations_proj, dist = 1320)

# Half-mile buffer
buffer_half <- st_buffer(cta_stations_proj, dist = 2640)

cat("Quarter-mile buffers created:", nrow(buffer_quarter), "\n")
```

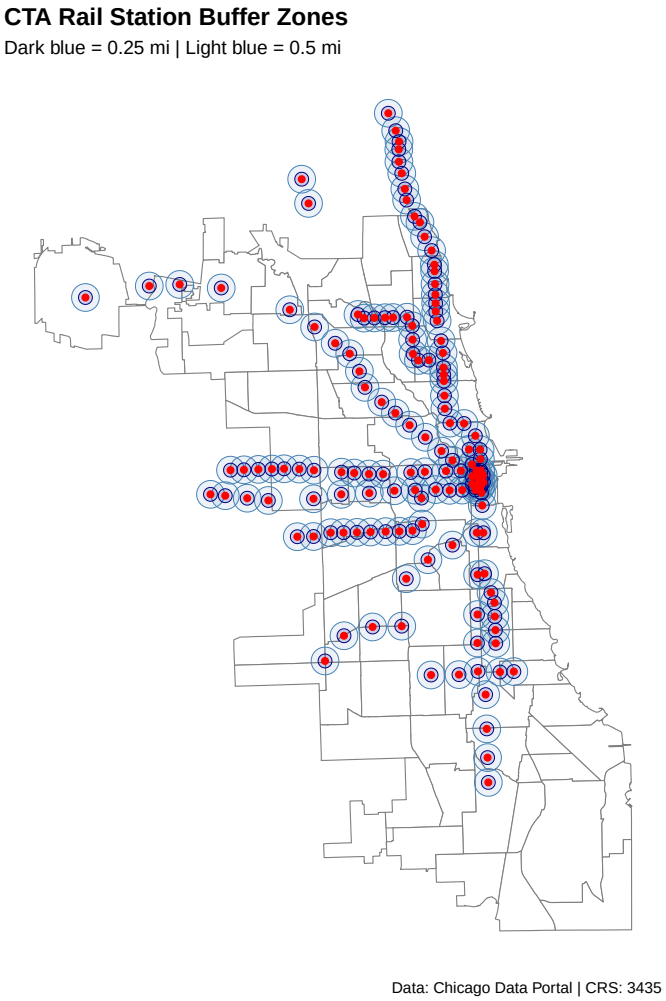
```
## Quarter-mile buffers created: 144
```

```
cat("Half-mile buffers created:", nrow(buffer_half), "\n")
```

```
## Half-mile buffers created: 144
```

Visualize the buffers

```
ggplot() +
  geom_sf(data = comm_areas_proj, fill = "white", color = "grey50", linewidth = 0.3) +
  geom_sf(data = buffer_half, fill = "steelblue", alpha = 0.1, color = "steelblue", linewidth = 0.2) +
  geom_sf(data = buffer_quarter, fill = "navy", alpha = 0.15, color = "navy", linewidth = 0.2) +
  geom_sf(data = cta_stations_proj, color = "red", size = 1.5) +
  labs(title = "CTA Rail Station Buffer Zones",
       subtitle = "Dark blue = 0.25 mi | Light blue = 0.5 mi",
       caption = "Data: Chicago Data Portal | CRS: 3435") +
  theme_void() +
  theme(plot.title = element_text(face = "bold", size = 14))
```



Step 5: Spatial Join — Count Crimes Within Each Buffer

```
# Count crimes within each station's quarter-mile buffer
crimes_in_quarter <- st_join(crime_proj, buffer_quarter, left = FALSE) %>%
  st_drop_geometry() %>%
  group_by(MAP_ID, STATION_NAME) %>%
  summarise(crimes_quarter_mi = n(), .groups = "drop")

# Count crimes within each station's half-mile buffer
crimes_in_half <- st_join(crime_proj, buffer_half, left = FALSE) %>%
  st_drop_geometry() %>%
  group_by(MAP_ID, STATION_NAME) %>%
  summarise(crimes_half_mi = n(), .groups = "drop")

cat("Stations with quarter-mile crime counts:", nrow(crimes_in_quarter), "\n")
```

```
## Stations with quarter-mile crime counts: 125
```

```
cat("Stations with half-mile crime counts:", nrow(crimes_in_half), "\n")
```

```
## Stations with half-mile crime counts: 128
```

Step 6: Merge Crime Counts with Ridership

```
# Join crime counts to station data
station_analysis <- cta_stations_proj %>%
  left_join(crimes_in_quarter, by = c("MAP_ID", "STATION_NAME")) %>%
  left_join(crimes_in_half, by = c("MAP_ID", "STATION_NAME")) %>%
  mutate(
    crimes_quarter_mi = replace_na(crimes_quarter_mi, 0),
    crimes_half_mi = replace_na(crimes_half_mi, 0)
  )

# Summary of the merged dataset
cat("\n=== Station Analysis Summary ===\n")
```

```
##
## === Station Analysis Summary ===
```

```
cat("Total stations:", nrow(station_analysis), "\n")
```

```
## Total stations: 144
```

```r
cat("Stations with ridership:", sum(!is.na(station_analysis$total_rides)), "\n")
```

```
## Stations with ridership: 143
```

```r
cat("Avg crimes (quarter-mile):", round(mean(station_analysis$crimes_quarter_mi)), "\n")
```

```
## Avg crimes (quarter-mile): 599
```

```r
cat("Avg crimes (half-mile):", round(mean(station_analysis$crimes_half_mi)), "\n")
```

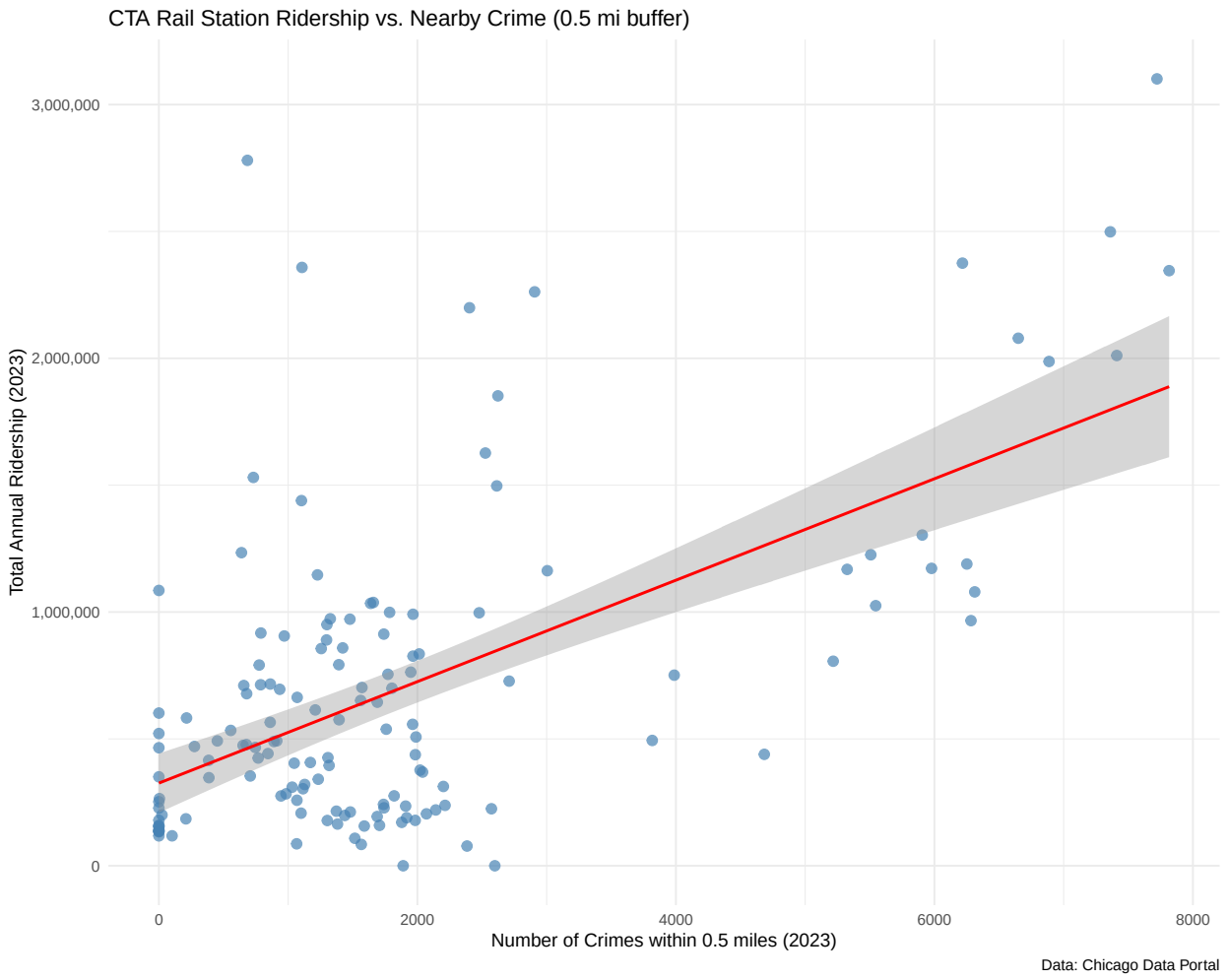
```
## Avg crimes (half-mile): 1859
```

Step 7: Non-Spatial EDA

7a. Scatter plot: Ridership vs Crime (half-mile buffer)

```r
station_df <- station_analysis %>%
  st_drop_geometry() %>%
  filter(!is.na(total_rides))

ggplot(station_df, aes(x = crimes_half_mi, y = total_rides)) +
  geom_point(color = "steelblue", alpha = 0.7, size = 2.5) +
  geom_smooth(method = "lm", se = TRUE, color = "red", linewidth = 0.8) +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "CTA Rail Station Ridership vs. Nearby Crime (0.5 mi buffer)",
    x = "Number of Crimes within 0.5 miles (2023)",
    y = "Total Annual Ridership (2023)",
    caption = "Data: Chicago Data Portal"
  ) +
  theme_minimal()
```

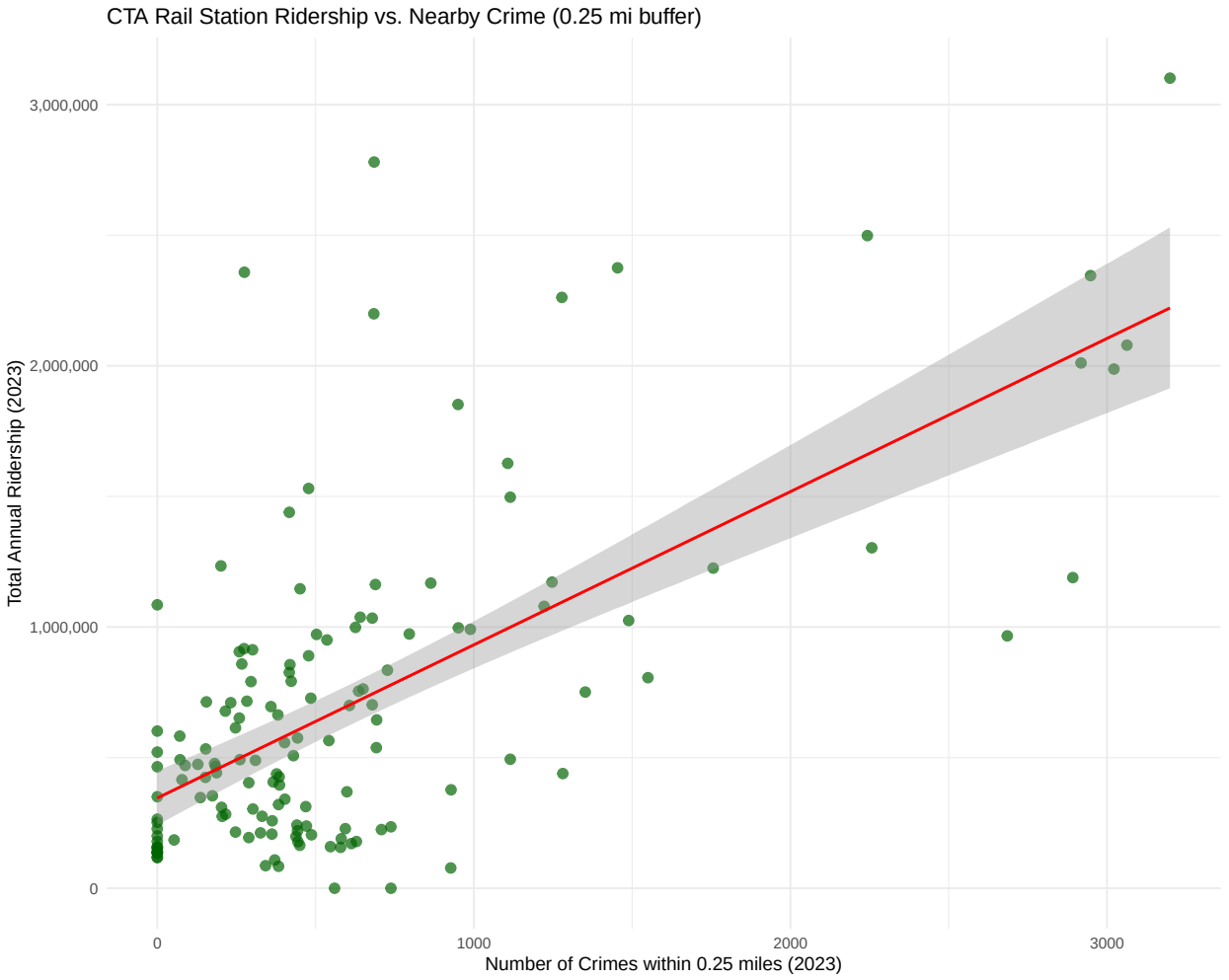



```
# Correlation
cor_val <- cor(station_df$crimes_half_mi, station_df$total_rides, use = "complete.obs")
cat("Pearson correlation (ridership ~ crimes_half_mi):", round(cor_val, 3), "\n")
```

```
## Pearson correlation (ridership ~ crimes_half_mi): 0.598
```

7b. Scatter plot: Ridership vs Crime (quarter-mile buffer)

```
ggplot(station_df, aes(x = crimes_quarter_mi, y = total_rides)) +
  geom_point(color = "darkgreen", alpha = 0.7, size = 2.5) +
  geom_smooth(method = "lm", se = TRUE, color = "red", linewidth = 0.8) +
  scale_y_continuous(labels = scales::comma) +
  labs(
    title = "CTA Rail Station Ridership vs. Nearby Crime (0.25 mi buffer)",
    x = "Number of Crimes within 0.25 miles (2023)",
    y = "Total Annual Ridership (2023)"
  ) +
  theme_minimal()
```

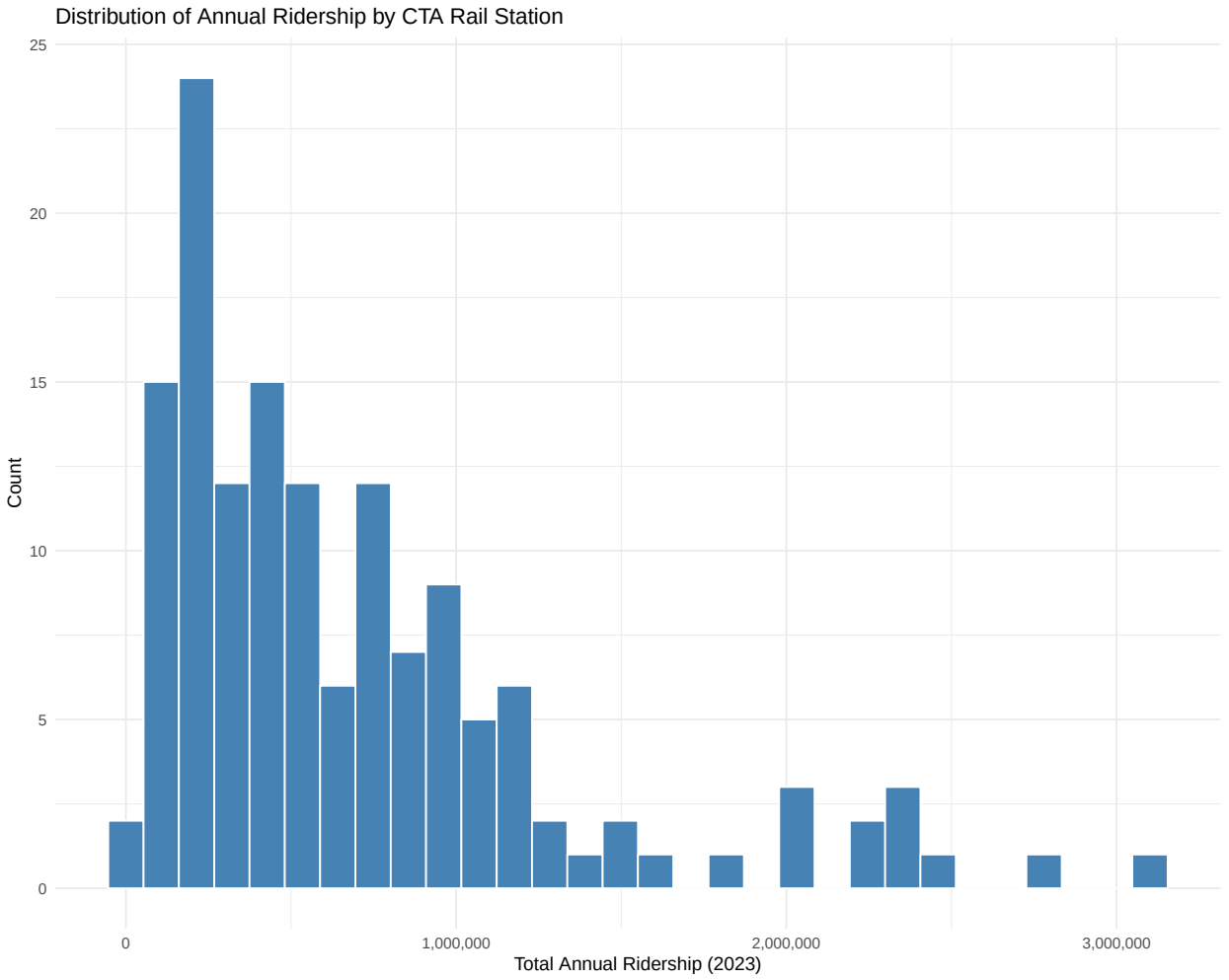



```
cor_val_q <- cor(station_df$crimes_quarter_mi, station_df$total_rides, use = "complete.obs")
cat("Pearson correlation (ridership ~ crimes_quarter_mi):", round(cor_val_q, 3), "\n")
```

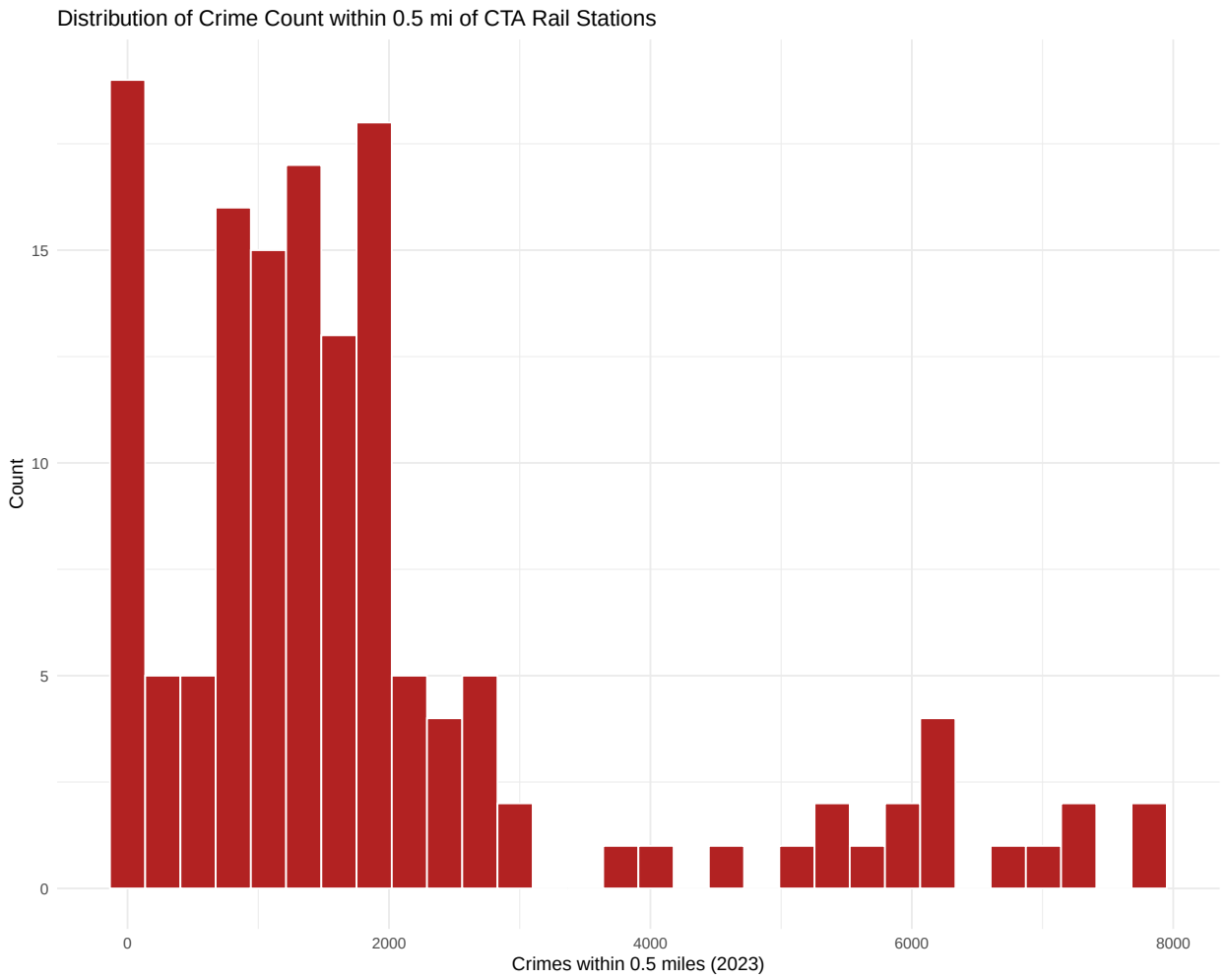
```
## Pearson correlation (ridership ~ crimes_quarter_mi): 0.649
```

7c. Distribution summaries

```
# Ridership distribution
ggplot(station_df, aes(x = total_rides)) +
  geom_histogram(fill = "steelblue", bins = 30, color = "white") +
  scale_x_continuous(labels = scales::comma) +
  labs(title = "Distribution of Annual Ridership by CTA Rail Station",
       x = "Total Annual Ridership (2023)", y = "Count") +
  theme_minimal()
```




```
# Crime exposure distribution
ggplot(station_df, aes(x = crimes_half_mi)) +
  geom_histogram(fill = "firebrick", bins = 30, color = "white") +
  labs(title = "Distribution of Crime Count within 0.5 mi of CTA Rail Stations",
       x = "Crimes within 0.5 miles (2023)", y = "Count") +
  theme_minimal()
```



Step 8: LISA Analysis — Crime Clustering by Census Tract

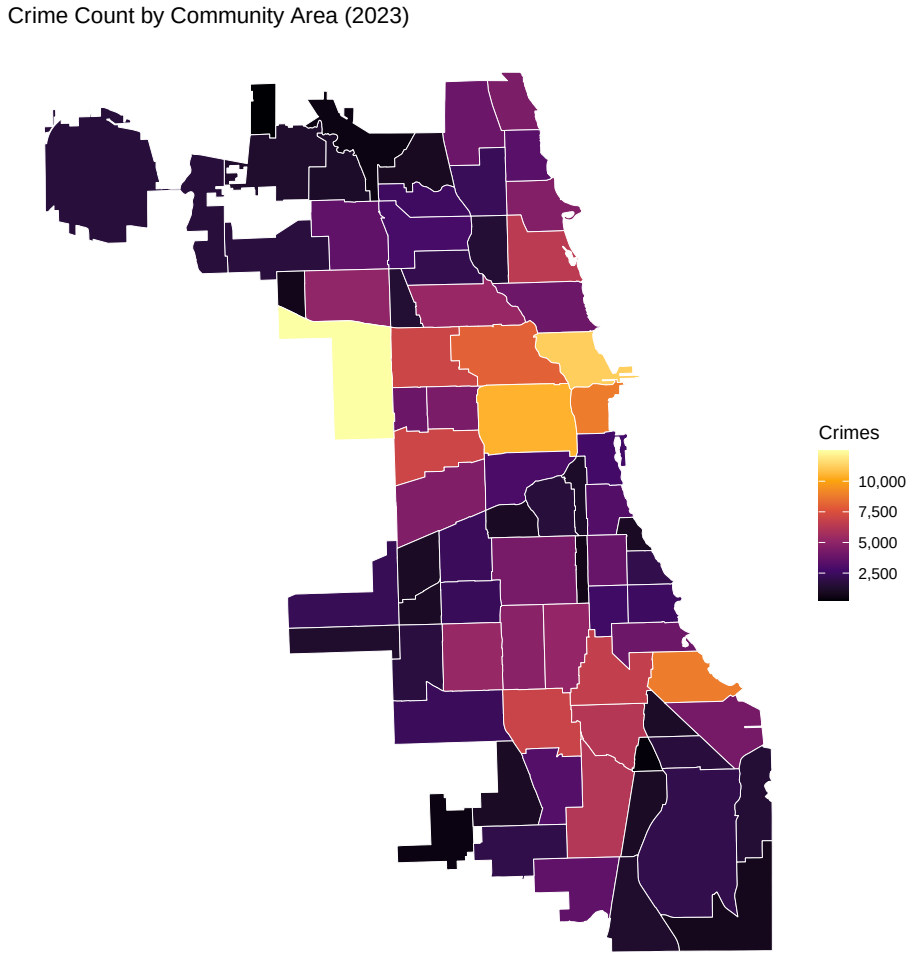
Per instructor feedback: explore whether crimes spatially cluster in Chicago using a LISA analysis by census tract (or community area).

```
# Count crimes per community area via spatial join
crime_by_ca <- st_join(crime_proj, comm_areas_proj, left = FALSE) %>%
  st_drop_geometry() %>%
  group_by(community) %>%
  summarise(crime_count = n(), .groups = "drop")

# Join back to community area polygons
comm_crime <- comm_areas_proj %>%
  left_join(crime_by_ca, by = "community") %>%
  mutate(crime_count = replace_na(crime_count, 0))

# Choropleth of crime by community area
ggplot(comm_crime) +
```

```
  geom_sf(aes(fill = crime_count), color = "white", linewidth = 0.2) +
  scale_fill_viridis_c(option = "inferno", labels = scales::comma) +
  labs(title = "Crime Count by Community Area (2023)",
       fill = "Crimes") +
  theme_void()
```



Compute spatial weights and LISA

```
# Create queen contiguity weights
comm_crime_valid <- comm_crime %>% filter(!is.na(crime_count))
nb <- poly2nb(comm_crime_valid, queen = TRUE)
lw <- nb2listw(nb, style = "W", zero.policy = TRUE)

# Global Moran's I
moran_result <- moran.test(comm_crime_valid$crime_count, lw, zero.policy = TRUE)
cat("Global Moran's I:", round(moran_result$estimate[1], 4), "\n")
```

```
## Global Moran's I: 0.3694
```

```r
cat("p-value:", moran_result$p.value, "\n")
```

```
## p-value: 3.755292e-08
```

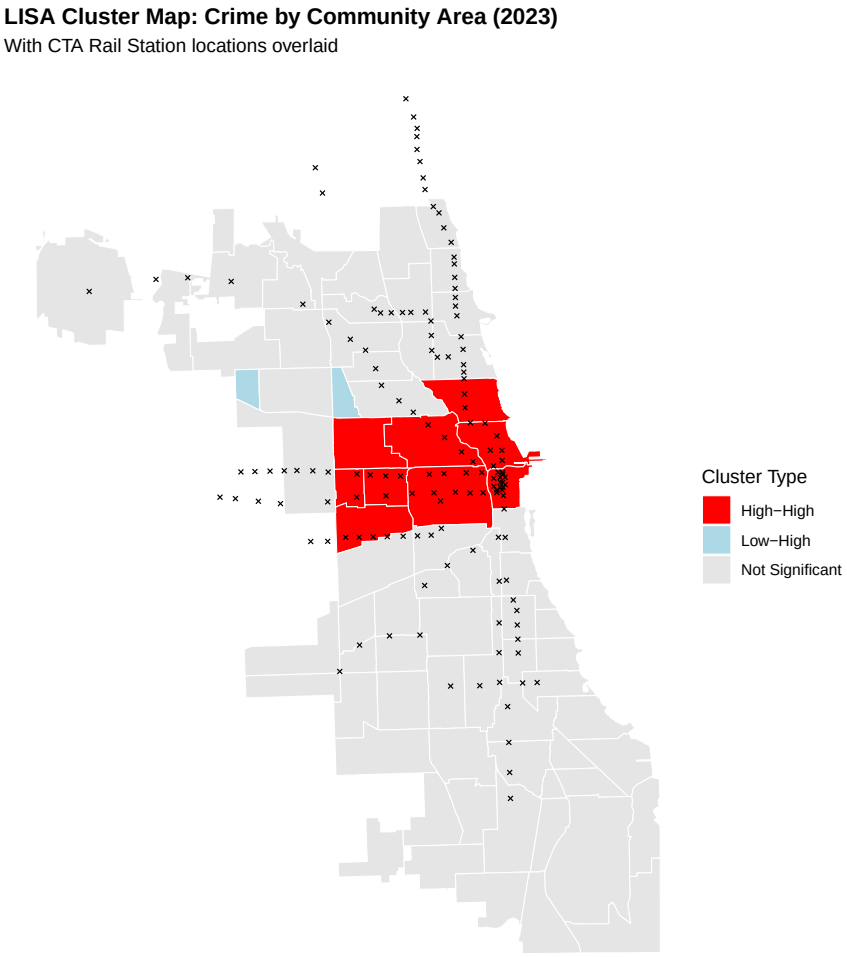
```r
# Local Moran's I (LISA)
lisa <- localmoran(comm_crime_valid$crime_count, lw, zero.policy = TRUE)

# Classify LISA clusters
comm_crime_valid$lisa_I <- lisa[, 1]
comm_crime_valid$lisa_p <- lisa[, 5]
mean_crime <- mean(comm_crime_valid$crime_count)
lag_crime <- lag.listw(lw, comm_crime_valid$crime_count, zero.policy = TRUE)

comm_crime_valid <- comm_crime_valid %>%
  mutate(
    lisa_cluster = case_when(
      lisa_p > 0.05 ~ "Not Significant",
      crime_count > mean_crime & lag_crime > mean_crime ~ "High-High",
      crime_count < mean_crime & lag_crime < mean_crime ~ "Low-Low",
      crime_count > mean_crime & lag_crime < mean_crime ~ "High-Low",
      crime_count < mean_crime & lag_crime > mean_crime ~ "Low-High",
      TRUE ~ "Not Significant"
    )
  )

# LISA cluster map
lisa_colors <- c("High-High" = "red", "Low-Low" = "blue",
                 "High-Low" = "pink", "Low-High" = "lightblue",
                 "Not Significant" = "grey90")

ggplot(comm_crime_valid) +
  geom_sf(aes(fill = lisa_cluster), color = "white", linewidth = 0.3) +
  scale_fill_manual(values = lisa_colors) +
  geom_sf(data = cta_stations_proj, color = "black", size = 1, shape = 4) +
  labs(title = "LISA Cluster Map: Crime by Community Area (2023)",
       subtitle = "With CTA Rail Station locations overlaid",
       fill = "Cluster Type") +
  theme_void() +
  theme(plot.title = element_text(face = "bold"))
```



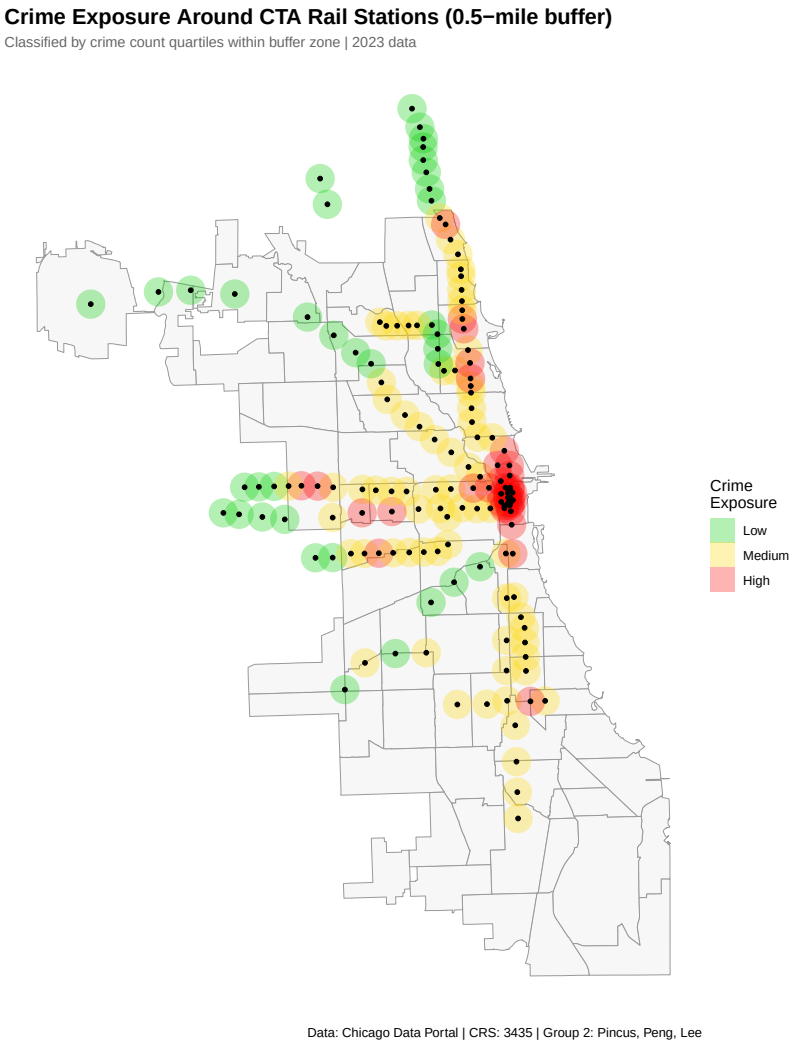
Step 9: Buffer Map Visualization — Crime Exposure Near Stations

This is the **main deliverable** showing crime density within station buffer zones.

```
# Classify stations by crime exposure level
station_analysis <- station_analysis %>%
  mutate(
    crime_level = case_when(
      crimes_half_mi >= quantile(crimes_half_mi, 0.75, na.rm = TRUE) ~ "High",
      crimes_half_mi >= quantile(crimes_half_mi, 0.25, na.rm = TRUE) ~ "Medium",
      TRUE ~ "Low"
    ),
    crime_level = factor(crime_level, levels = c("Low", "Medium", "High"))
  )

# Recreate buffers with the classified data
buffer_map <- st_buffer(station_analysis, dist = 2640)
```

```r
ggplot() +
  geom_sf(data = comm_areas_proj, fill = "grey97", color = "grey60", linewidth = 0.3) +
  geom_sf(data = buffer_map, aes(fill = crime_level), alpha = 0.3, color = NA) +
  scale_fill_manual(values = c("Low" = "green3", "Medium" = "gold", "High" = "red")) +
  geom_sf(data = cta_stations_proj, color = "black", size = 1.2) +
  labs(
    title = "Crime Exposure Around CTA Rail Stations (0.5-mile buffer)",
    subtitle = "Classified by crime count quartiles within buffer zone | 2023 data",
    fill = "Crime\nExposure",
    caption = "Data: Chicago Data Portal | CRS: 3435 | Group 2: Pincus, Peng, Lee"
  ) +
  theme_void() +
  theme(
    plot.title = element_text(face = "bold", size = 15),
    plot.subtitle = element_text(size = 10, color = "grey40"),
    legend.position = "right"
  )
```



```
ggsave("PBL_Buffer_Crime_Map.png", width = 12, height = 10, dpi = 300, bg = "white")
```

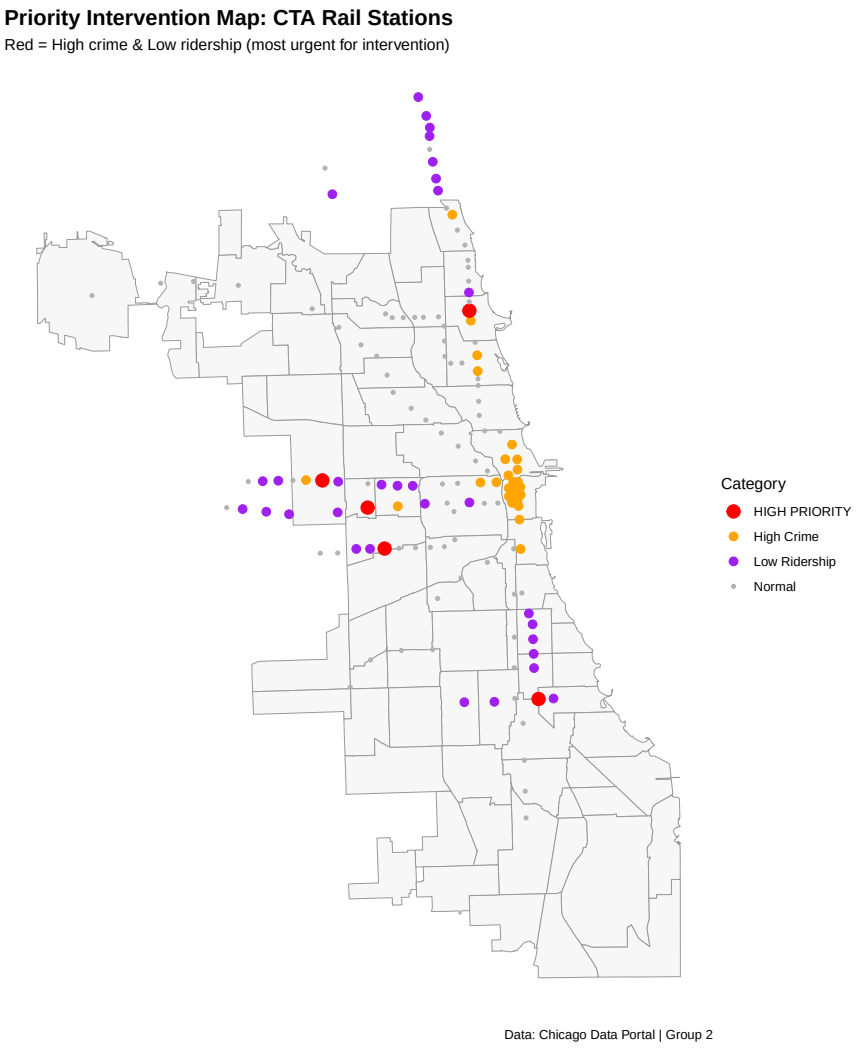
Step 10: Priority Intervention Map

Identify stations that are both **high-crime** and **low-ridership** — these are the most urgent candidates for intervention.

```
station_analysis <- station_analysis %>%
  mutate(
    ridership_level = case_when(
      is.na(total_rides) ~ NA_character_,
      total_rides <= quantile(total_rides, 0.25, na.rm = TRUE) ~ "Low Ridership",
      TRUE ~ "Other"
    ),
    priority = case_when(
      crime_level == "High" & ridership_level == "Low Ridership" ~ "HIGH PRIORITY",
      crime_level == "High" ~ "High Crime",
      ridership_level == "Low Ridership" ~ "Low Ridership",
      TRUE ~ "Normal"
    ),
    priority = factor(priority, levels = c("HIGH PRIORITY", "High Crime",
                                           "Low Ridership", "Normal"))
  )

priority_colors <- c("HIGH PRIORITY" = "red", "High Crime" = "orange",
                     "Low Ridership" = "purple", "Normal" = "grey70")

ggplot() +
  geom_sf(data = comm_areas_proj, fill = "grey97", color = "grey60", linewidth = 0.3) +
  geom_sf(data = station_analysis, aes(color = priority, size = priority)) +
  scale_color_manual(values = priority_colors) +
  scale_size_manual(values = c("HIGH PRIORITY" = 4, "High Crime" = 2.5,
                               "Low Ridership" = 2.5, "Normal" = 1)) +
  labs(
    title = "Priority Intervention Map: CTA Rail Stations",
    subtitle = "Red = High crime & Low ridership (most urgent for intervention)",
    color = "Category", size = "Category",
    caption = "Data: Chicago Data Portal | Group 2"
  ) +
  theme_void() +
  theme(plot.title = element_text(face = "bold", size = 15))
```




```
ggsave("PBL_Priority_Map.png", width = 12, height = 10, dpi = 300, bg = "white")

# Print the priority stations
cat("\n=== HIGH PRIORITY STATIONS ===\n")
```

```
##
## === HIGH PRIORITY STATIONS ===
```

```
station_analysis %>%
  st_drop_geometry() %>%
  filter(priority == "HIGH PRIORITY") %>%
  select(STATION_NAME, total_rides, avg_daily_rides, crimes_quarter_mi, crimes_half_mi) %>%
  arrange(desc(crimes_half_mi)) %>%
  print()
```

```
## # A tibble: 5 x 5
##   STATION_NAME total_rides avg_daily_rides crimes_quarter_mi crimes_half_mi
##   <chr>              <int>           <dbl>             <int>          <int>
## 1 Lawrence               0               0               738           2599
```

```
## 2 Pulaski           224656          615.            708          2573
## 3 King Drive         77755          213.            927          2384
## 4 Central Park      219936          603.            443          2142
## 5 Laramie           204646          561.            487          2070
```

Step 11: Supplementary Regression

Simple OLS to test the relationship (supplementary to the spatial analysis focus).

```r
reg_df <- station_analysis %>%
  st_drop_geometry() %>%
  filter(!is.na(total_rides))

model <- lm(total_rides ~ crimes_half_mi, data = reg_df)
summary(model)
```

```
##
## Call:
## lm(formula = total_rides ~ crimes_half_mi, data = reg_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -845361 -350819  -73464  224164 2317455
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    325810.49   59051.59   5.517 1.60e-07 ***
## crimes_half_mi    199.90      22.57   8.856 3.18e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 495800 on 141 degrees of freedom
## Multiple R-squared:  0.3574, Adjusted R-squared:  0.3529
## F-statistic: 78.43 on 1 and 141 DF,  p-value: 3.183e-15
```

```r
# Log-transformed version (ridership is likely right-skewed)
reg_df <- reg_df %>%
  mutate(log_rides = log(total_rides + 1),
         log_crimes = log(crimes_half_mi + 1))

model_log <- lm(log_rides ~ log_crimes, data = reg_df)
summary(model_log)
```

```
##
## Call:
## lm(formula = log_rides ~ log_crimes, data = reg_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -13.0932  -0.4004   0.2014   0.7364   1.8883
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12.24605    0.40838  29.987   <2e-16 ***
## log_crimes   0.10774    0.05891   1.829   0.0695 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.748 on 141 degrees of freedom
## Multiple R-squared:  0.02318,    Adjusted R-squared:  0.01625
## F-statistic: 3.345 on 1 and 141 DF,  p-value: 0.06951
```

Step 12: Police Station Proximity (Context)

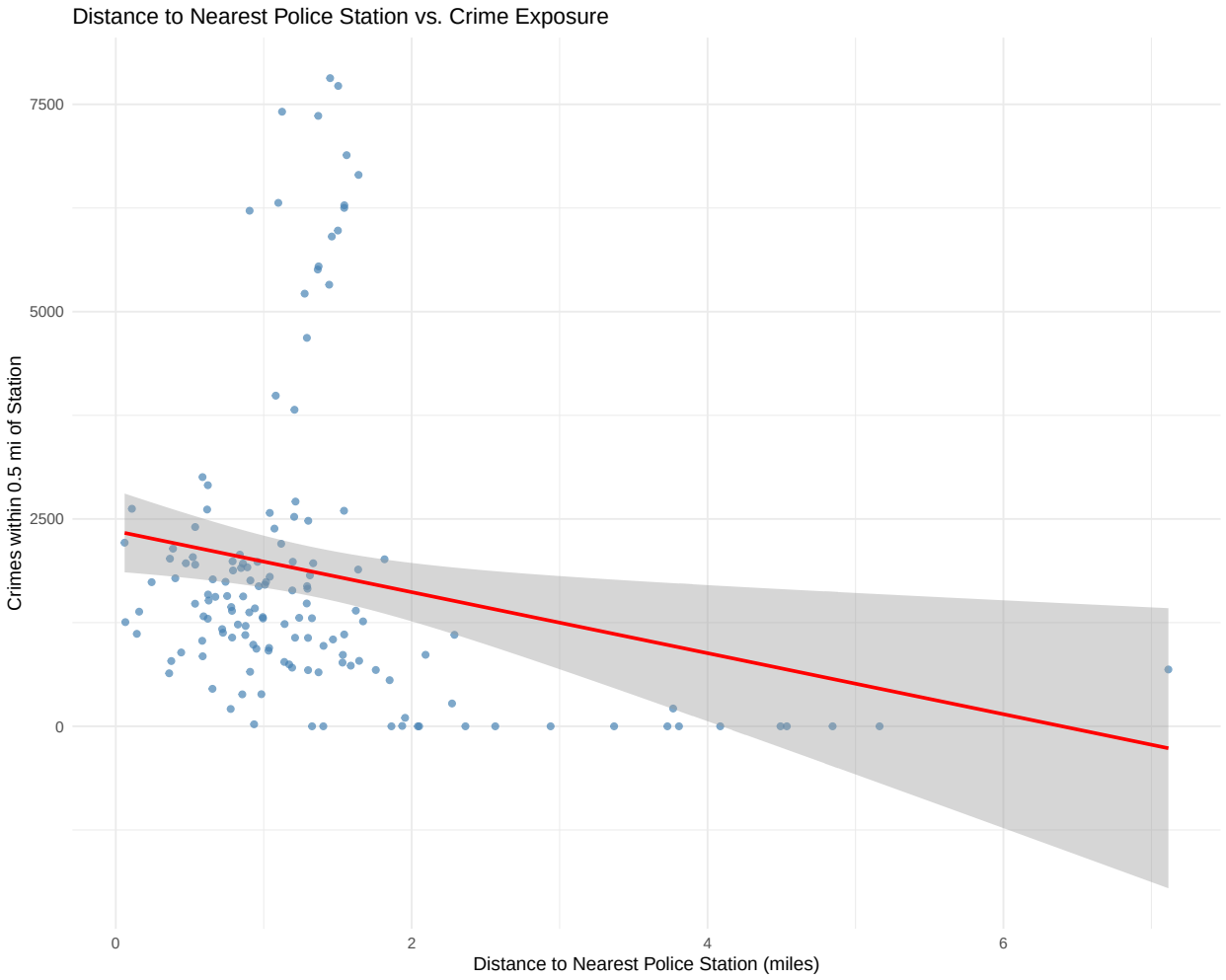
```
# If police stations were loaded, check distance from each CTA station to nearest police station
if (exists("police_proj")) {
  # Nearest police station distance for each CTA rail station
  nearest_idx <- st_nearest_feature(cta_stations_proj, police_proj)
  nearest_dist <- st_distance(cta_stations_proj, police_proj[nearest_idx, ], by_element = TRUE)

  station_analysis$dist_to_police_ft <- as.numeric(nearest_dist)
  station_analysis$dist_to_police_mi <- station_analysis$dist_to_police_ft / 5280

  cat("Average distance to nearest police station:",
      round(mean(station_analysis$dist_to_police_mi, na.rm = TRUE), 2), "miles\n")

  # Scatter: police distance vs crime
  police_df <- station_analysis %>% st_drop_geometry()
  ggplot(police_df, aes(x = dist_to_police_mi, y = crimes_half_mi)) +
    geom_point(color = "steelblue", alpha = 0.7) +
    geom_smooth(method = "lm", se = TRUE, color = "red") +
    labs(title = "Distance to Nearest Police Station vs. Crime Exposure",
         x = "Distance to Nearest Police Station (miles)",
         y = "Crimes within 0.5 mi of Station") +
    theme_minimal()
}
```

```
## Average distance to nearest police station: 1.34 miles
```



Summary & Key Takeaways

```
# Final summary table
summary_table <- station_analysis %>%
  st_drop_geometry() %>%
  filter(!is.na(total_rides)) %>%
  select(STATION_NAME, total_rides, avg_daily_rides,
         crimes_quarter_mi, crimes_half_mi, priority) %>%
  arrange(desc(crimes_half_mi))

cat("\n=== TOP 15 STATIONS BY CRIME EXPOSURE (0.5 mi) ===\n")
```

```
##
## === TOP 15 STATIONS BY CRIME EXPOSURE (0.5 mi) ===
```

```r
print(head(summary_table, 15))
```

```
## # A tibble: 15 x 6
##    STATION_NAME     total_rides avg_daily_rides crimes_quarter_mi crimes_half_mi
##    <chr>                  <int>           <dbl>              <int>          <int>
##  1 State/Lake           2345432           6426.               2948           7816
##  2 Lake                 3101354           8497.               3199           7722
##  3 Grand                2010967           5509.               2918           7411
##  4 Clark/Lake           2498459           6845.               2243           7361
##  5 Washington           1987482           5445.               3022           6887
##  6 Washington/Waba~     2079125           5696.               3063           6649
##  7 Merchandise Mart     1079411           2957.               1222           6312
##  8 Monroe                966150           2647.               2685           6283
##  9 Monroe               1189272           3258.               2892           6251
## 10 Chicago              2375146           6507.               1454           6217
## 11 Washington/Wells     1172095           3211.               1247           5978
## 12 Adams/Wabash         1303464           3571.               2257           5906
## 13 Jackson              1025052           2808.               1489           5546
## 14 Jackson              1225612           3358.               1756           5508
## 15 Quincy/Wells         1168184           3201.                864           5325
## # i 1 more variable: priority <fct>
```

```r
cat("\n=== BOTTOM 15 STATIONS BY CRIME EXPOSURE ===\n")
```

```
##
## === BOTTOM 15 STATIONS BY CRIME EXPOSURE ===
```

```r
print(tail(summary_table, 15))
```

```
## # A tibble: 15 x 6
##    STATION_NAME    total_rides avg_daily_rides crimes_quarter_mi crimes_half_mi
##    <chr>                 <int>           <dbl>              <int>          <int>
##  1 Davis                521222           1428.                  0              0
##  2 Dempster-Skokie      252346            691.                  0              0
##  3 Oak Park             152748            418.                  0              0
##  4 Main                 179436            492.                  0              0
##  5 Forest Park          350494            960.                  0              0
##  6 Noyes                156904            430.                  0              0
##  7 Foster               137875            378.                  0              0
##  8 54th/Cermak          465138           1274.                  0              0
##  9 Dempster             141099            387.                  0              0
## 10 Rosemont            1085136           2973.                  0              0
## 11 Harlem               117854            323.                  0              0
## 12 Linden               159186            436.                  0              0
## 13 Central              135516            371.                  0              0
## 14 Oak Park             227528            623.                  0              0
## 15 Oakton-Skokie        133944            367.                  0              0
## # i 1 more variable: priority <fct>
```

Session Info

```
sessionInfo()
```

```
## R version 4.5.2 (2025-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 26200)
##
## Matrix products: default
##   LAPACK version 3.12.1
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Chicago
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] viridis_0.6.5     viridisLite_0.4.2 tidyr_1.3.2       jsonlite_2.0.0
##  [5] spdep_1.4-2       spData_2.3.4      tmap_4.2          ggplot2_4.0.1
##  [9] dplyr_1.1.4       sf_1.0-23
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1   farver_2.1.2       S7_0.2.1           fastmap_1.2.0
##  [5] leaflegend_1.2.1   leaflet_2.2.3      XML_3.99-0.20      digest_0.6.39
##  [9] lifecycle_1.0.4    terra_1.8-86       magrittr_2.0.4     compiler_4.5.2
## [13] rlang_1.1.6        tools_4.5.2        utf8_1.2.6         yaml_2.3.12
## [17] data.table_1.18.0  knitr_1.51         labeling_0.4.3     htmlwidgets_1.6.4
## [21] sp_2.2-0           classInt_0.4-11    RColorBrewer_1.1-3 abind_1.4-8
## [25] KernSmooth_2.23-26 withr_3.0.2        purrr_1.2.0        leafsync_0.1.0
## [29] grid_4.5.2         cols4all_0.10      e1071_1.7-17       leafem_0.2.5
## [33] colorspace_2.1-2   spacesXYZ_1.6-0    scales_1.4.0       cli_3.6.5
## [37] rmarkdown_2.30     ragg_1.5.0         generics_0.1.4     otel_0.2.0
## [41] rstudioapi_0.17.1  tmaptools_3.3      DBI_1.2.3          proxy_0.4-29
## [45] splines_4.5.2      stars_0.7-0        parallel_4.5.2     s2_1.1.9
## [49] base64enc_0.1-3    vctrs_0.6.5        boot_1.3-32        Matrix_1.7-4
## [53] systemfonts_1.3.1  crosstalk_1.2.2    units_1.0-0        maptiles_0.11.0
## [57] glue_1.8.0         lwgeom_0.2-14      codetools_0.2-20   gtable_0.3.6
## [61] deldir_2.0-4       raster_3.6-32      tibble_3.3.0       logger_0.4.1
## [65] pillar_1.11.1      htmltools_0.5.9    R6_2.6.1           textshaping_1.0.4
## [69] wk_0.9.5           evaluate_1.0.5     lattice_0.22-7     png_0.1-8
## [73] class_7.3-23       Rcpp_1.1.0         gridExtra_2.3      nlme_3.1-168
## [77] mgcv_1.9-3         xfun_0.55          pkgconfig_2.0.3
```