DataSci 306, Homework 3

Max Han, maxhan

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
```

Traffic Crash Analysis

We will explore Detroit City traffic crash patterns in this homework

```
df <- read_csv("./data/Traffic_Crashes.csv")
```

```
## Rows: 267078 Columns: 52
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): primary_road, intersecting_road, crash_date, surface_type
## dbl (25): X, Y, OID, crash_id, day, month, year, hour, weekday, crash_type_c...
## lgl (23): community_code, is_property_damage_only, is_secondary_crash, is_tr...
##
## 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.
```

```
df |> glimpse()
```

```
## Rows: 267,078
## Columns: 52
## $ X                          <dbl> -83.2142, -83.0693, -83.1119, -83.2773,~
## $ Y                          <dbl> 42.4012, 42.3462, 42.3459, 42.4120, 42.~
## $ OID                        <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, ~
## $ crash_id                   <dbl> 7941954, 7851614, 7847628, 7988406, 784~
## $ primary_road               <chr> "FENKELL", "N M 10", "W/B I-94 FWY", "T~
## $ intersecting_road          <chr> "OAKFIELD", "MARTIN LUTHER KING JR BLVD~
## $ crash_date                 <chr> "2011/01/01 05:00:00+00", "2011/01/01 0~
## $ day                        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ month                      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ year                       <dbl> 2011, 2011, 2011, 2011, 2011, 2011, 201~
```

```
## $ hour                            <dbl> 22, 3, 2, 1, 8, 3, 5, 3, 6, 4, 23, 2, 1~
## $ weekday                         <dbl> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, ~
## $ crash_type_code                 <dbl> 5, 5, 10, 4, 1, 9, 4, 4, 5, 5, 5, 1, 0,~
## $ highway_classification_code     <dbl> 9, 3, 1, 2, 1, 9, 3, 9, 1, 1, 9, 9, 9, ~
## $ community_code                  <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ jurisdiction_code               <dbl> 2, 1, 1, 5, 1, 5, 1, 5, 1, 1, 4, 4, 4, ~
## $ lane_departure_type_code        <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, ~
## $ surface_type                    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ road_condition_code             <dbl> 1, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 8, 1, ~
## $ weather_condition_code          <dbl> 1, 4, 4, 4, 2, 4, 4, 2, 4, 4, 1, 1, 1, ~
## $ lighting_condition_code         <dbl> 3, 4, 4, 4, 1, 4, 4, 4, 4, 4, 4, 4, 1, ~
## $ speed_limit                     <dbl> 30, 55, 55, 45, 70, 25, 30, 25, 55, 55,~
## $ num_lanes                       <dbl> 5, 3, 3, 4, 4, 2, 7, 2, 3, 4, 2, 3, 2, ~
## $ num_units                       <dbl> 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2, ~
## $ num_occupants                   <dbl> 2, 7, 4, 3, 1, 4, 5, 3, 2, 1, 5, 1, 2, ~
## $ most_severe_injury_code         <dbl> 5, 5, 5, 5, 5, 5, 4, 5, 4, 4, 5, 5, 5, ~
## $ num_fatal_injuries              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ num_suspected_serious_injuries <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ num_suspected_minor_injuries    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ num_possible_injuries           <dbl> 0, 0, 0, 0, 0, 0, 1, 0, 2, 1, 0, 0, 0, ~
## $ is_property_damage_only         <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FAL~
## $ is_secondary_crash              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_traffic_control_disregarded <lgl> FALSE, FALSE, FALSE, TRUE, FALSE, FALSE~
## $ is_red_light_run_involved       <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_hit_and_run_involved         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, TRUE~
## $ is_alcohol_involved             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_drug_involved                <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_unbelted_person_involved     <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_work_zone_involved           <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_speeding_driver_involved     <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_distracted_driver_involved   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_driveway_involved            <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_pedestrian_involved          <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_elderly_driver_involved      <lgl> FALSE, FALSE, TRUE, FALSE, FALSE, FALSE~
## $ is_young_driver_involved        <lgl> FALSE, FALSE, TRUE, TRUE, TRUE, FALSE, ~
## $ is_commercial_vehicle_involved <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_emergency_vehicle_involved   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ is_train_involved               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_school_bus_involved          <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_motorcycle_involved          <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_bicycle_involved             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ is_deer_involved                <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
```

Source of data: https://data.detroitmi.gov/datasets/d837b05bdd9643698be30dfedbab0272

This dataset pertains to traffic crashes that occurred within the City of Detroit from 2011-2022.

Question 1 (1.5 points)

During which periods of the day are accidents most likely to happen?

Before we answer this question, let us revisit the `cut` function again:

The `cut` function is useful for breaking up a quantitative variable into discrete categories. Here is an example:

```
x <- c(2, -1, 0.5, 10, 3, 4, -0.25, 6)
cut(x, breaks = c(-Inf, 0, 5, Inf), labels = c("Small", "Medium", "Large"))
```

```
## [1] Medium Small  Medium Large  Medium Medium Small  Large
## Levels: Small Medium Large
```

The notation `(a, b]` means that the interval is defined by $a < x \leq b$ (i.e., a half closed interval).

Note, you can give the same label twice to get two cuts to be the same group.

```
cut(x, breaks = c(-Inf, 0, 5, Inf), labels = c("A", "B", "A"))
```

```
## [1] B A B A B B A A
## Levels: A B
```

Using the `cut` function and `mutate`, make a new column that breaks the day into the following periods:

- Work Hours: after 10am and until 5pm. i.e., (10am - 5pm]
- Non-Work Hours: after 6am and until 10am and after 5pm and until 8pm. i.e., (6am to 10am] and (5pm to 8pm]
- Night: after 8pm and until 6am. i.e., (8pm to 6am]

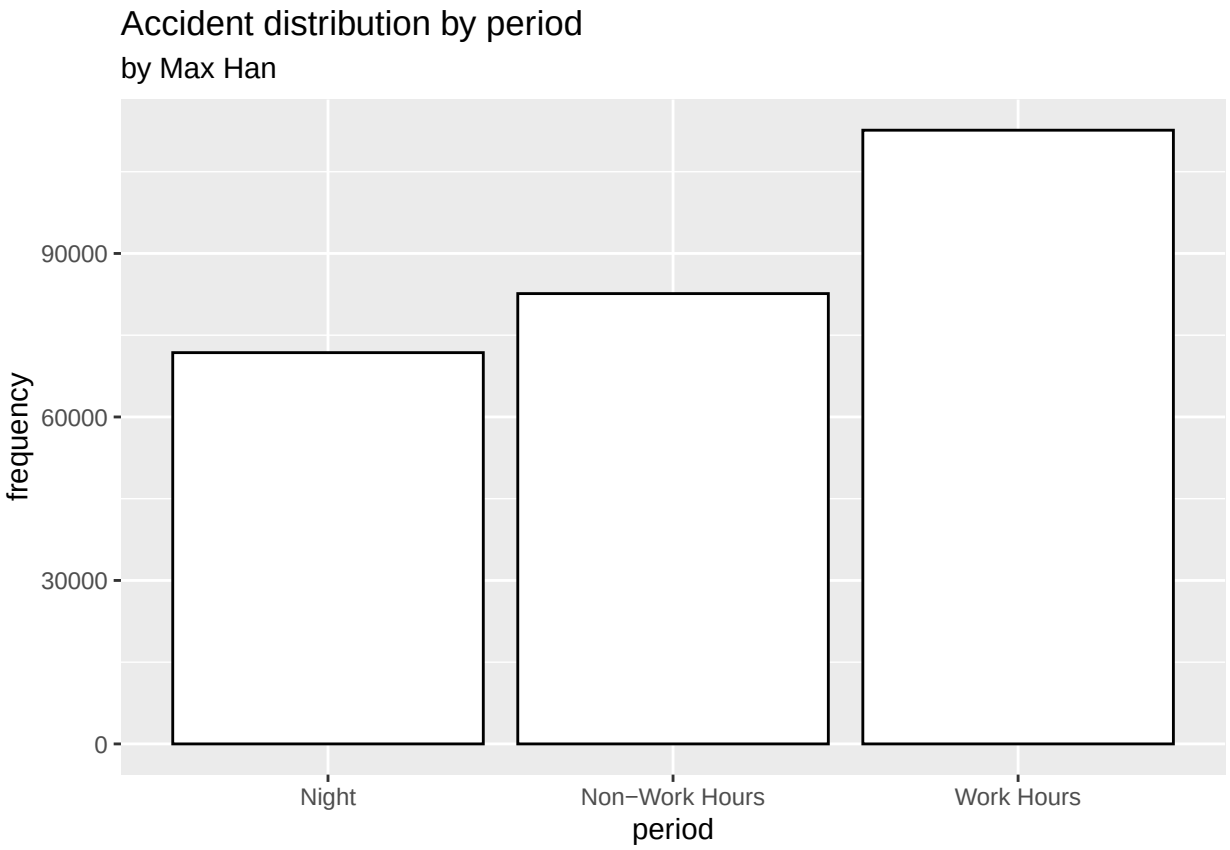
Hints: `hour` column in the data set is on a 24 clock. To capture all the observations, make the first `breaks` value strictly less than 0

Filter out the period's that are NA, if any, and then plot the distribution of accidents in various period's of the day as a bar chart.

```
df1 <- df |> mutate(df, period = cut(hour, breaks = c(-Inf, 6, 10, 17, 20, Inf),
                                     labels = c("Night","Non-Work Hours", "Work Hours",
                                                "Non-Work Hours", "Night")),
             .before = hour)
select(df1, period, hour)
```

```
## # A tibble: 267,078 x 2
##    period          hour
##    <fct>          <dbl>
##  1 Night             22
##  2 Night              3
##  3 Night              2
##  4 Night              1
##  5 Non-Work Hours     8
##  6 Night              3
##  7 Night              5
##  8 Night              3
##  9 Night              6
## 10 Night              4
## # i 267,068 more rows
```

```
df2 <- df1 |> filter(!is.na(period))
df2 |> ggplot(aes(x = period)) +
  geom_bar(color = "black", fill = "white") +
  labs(title = "Accident distribution by period",
       subtitle = "by Max Han",
       x = "period",
       y = "frequency")
```



Question 2 (2 points)

The most dangerous intersecting roads and their crash trends

Identify the three intersecting roads with the highest number of accidents. You can get this count by grouping on `intersecting_road`. Then, create a line chart that displays the annual (use the `year` column to group) accident count for each of these intersecting roads across the entire dataset. All three lines should be plotted on the same chart for comparison.

Based on the observed trend, can you speculate on potential reasons for its shape? Feel free to explore hypothetical explanations, given the limited information available.

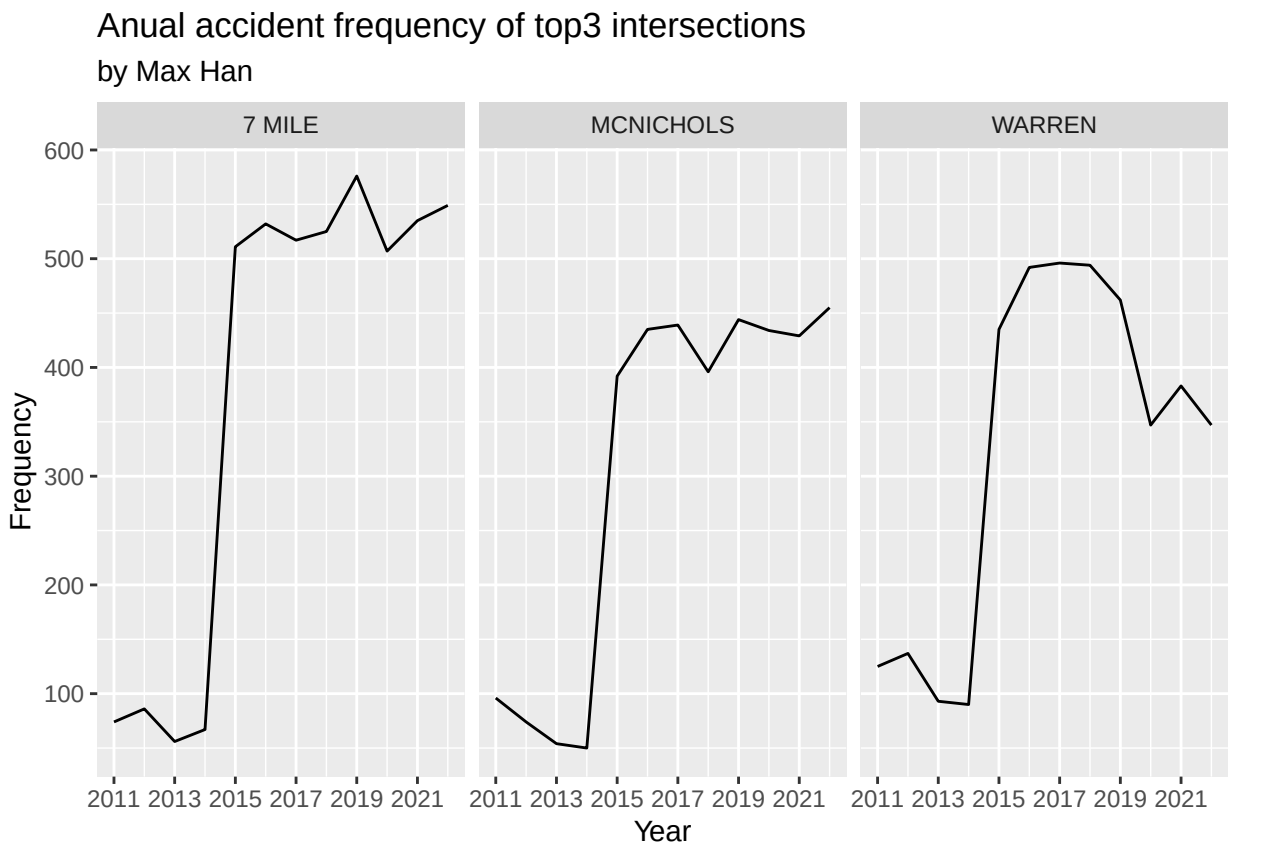
Hint: Ensure the year labels on your chart do not show decimal points for full credit.

```
df2 |> group_by(intersecting_road) |> summarize(n = n()) |> slice_max(n = 3, n) -> top3
top3
```

```
## # A tibble: 3 x 2
##   intersecting_road     n
##   <chr>             <int>
## 1 7 MILE             4535
## 2 WARREN             3901
## 3 MCNICHOLS          3698
```

```
df2 |> filter(intersecting_road %in% c("7 MILE", "WARREN", "MCNICHOLS")) |>
  group_by(intersecting_road, year) |> summarize(n = n(), .groups = "drop") |>
  ggplot(aes(x = year, y = n)) +
  geom_line() +
  facet_wrap(~intersecting_road) +
```

```
  scale_x_continuous(breaks = seq(floor(min(df2$year)), ceiling(max(df2$year)), by = 2)) +
  labs(title = "Anual accident frequency of top3 intersections",
       subtitle = "by Max Han",
       x = "Year",
       y = "Frequency")
```



The three line charts shows a rapid accident in crease from 2013 to 2015, which implies some big change in Detroid area.

Question 3 (1 point)

a) Finding the weekday (0.5)

There is a `weekday` column in this dataset. Which weekday does number 7 correspond to in this dataset? No need to write code to find this answer. You may use your computer calendar to figure this out.

Number 7 stands for Sunday.

b) The most dangerous weekday (0.5 point)

Which week day has the highest number of accidents? Saturday has the highest number of accidents.

```
df2 |> group_by(weekday) |> summarize(n = n()) |> slice_max(n = 3, n)
```

```
## # A tibble: 3 x 2
##   weekday     n
##     <dbl> <int>
## 1       6 42316
## 2       7 39361
## 3       5 38614
```

Question 4 (1 point)

Investigating speed_limit

Identify and count the number of records in a dataset that contain invalid speed limit values. These invalid values include:

- Speed limits of 0.
- Speed limits that are not multiples of 5.
- Speed limits exceeding the legal maximum of 70 in Michigan.

```
df |> filter(speed_limit == 0 | speed_limit %% 5 != 0 | speed_limit > 70) |> count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 13763
```

Question 5 (2.5 point)

a) How many crashes involve both young and old? (0.5 point)

Find the number of records that has both young driver and elderly driver involved in the same accident.

```
df |> filter(is_elderly_driver_involved & is_young_driver_involved) |> count()
```

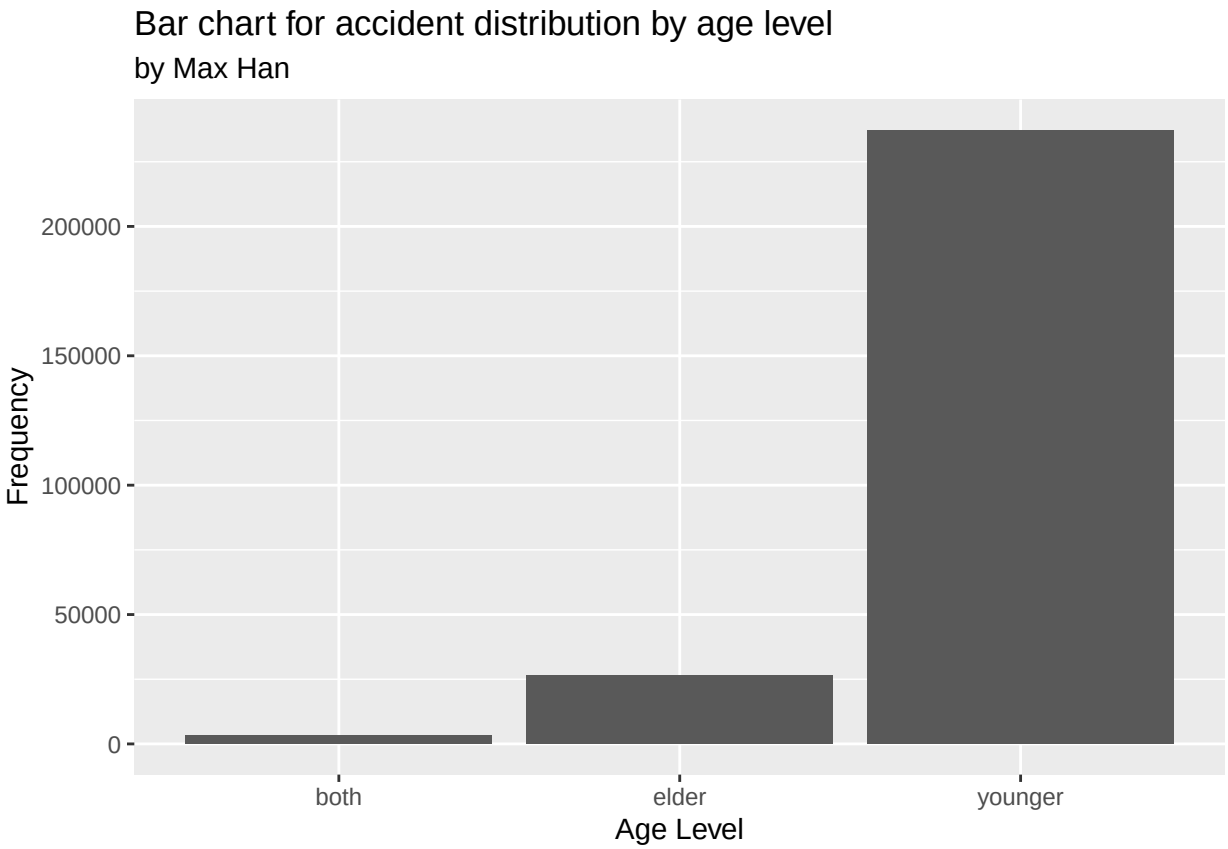
```
## # A tibble: 1 x 1
##       n
##   <int>
## 1  3252
```

b) Distribution (2 points)

Plot the distribution of crashes due to only young drivers, only elderly drivers and both. Do not show other records.

Hint: One way to do is to mutate a new column that denotes the labels as Both, Young and Elderly based on the values in respective columns and then use this new column to plot

```
df3 <- mutate(df, age_level = if_else(is_elderly_driver_involved & is_young_driver_involved, "both",
                                      if_else(is_elderly_driver_involved, "elder", "younger")))
ggplot(data = df3, aes(x = age_level)) +
geom_bar() +
  labs(title = "Bar chart for accident distribution by age level",
       subtitle = "by Max Han",
       x = "Age Level",
       y = "Frequency")
```



Challenge problem (2 pt)

Fixing erroneous speed_limit values based on other records

Most of the anomalies in speed_limit could be due to data entry errors. We will try to fix as many as possible by finding the most commonly occurring speed_limit (in other words the 'mode') for the same intersecting_road and primary_road combination. Create a new column called 'corrected_speed_limit' and place the mode value that you derived from the data into this new column. Then count the records that still have '0', '90', and '95' in the corrected_speed_limit column and display the total count for each of them.

Hint: There may be no ready function available for finding `mode`. You may have to define one.

```
# Function to find the mode
get_mode <- function(v) {
  uniq_vals <- unique(v)
  uniq_vals[which.max(tabulate(match(v, uniq_vals)))]
}

# Group by 'intersecting_road' and 'primary_road', and find the mode of 'speed_limit'
df <- df %>%
  group_by(intersecting_road, primary_road) %>%
  mutate(corrected_speed_limit = get_mode(speed_limit)) %>%
  ungroup()

df |> select(intersecting_road, primary_road, speed_limit, corrected_speed_limit)
```

```
## # A tibble: 267,078 x 4
##    intersecting_road          primary_road speed_limit corrected_speed_limit
```

```
##    <chr>                      <chr>              <dbl>                 <dbl>
##  1 OAKFIELD                   FENKELL               30                    35
##  2 MARTIN LUTHER KING JR BLVD N M 10                55                    55
##  3 SCOTTEN RD                 W/B I-94 FWY          55                    55
##  4 GROVE                      TELEGRAPH             45                    45
##  5 NEVADA                     I-75 FWY              70                    70
##  6 VASSAR                     SNOWDEN               25                    25
##  7 E WILLIS                   WOODWARD AVE          30                    30
##  8 W MCNICHOLS                WASHBURN              25                    25
##  9 VAN DYKE AVE               I 94                  55                    55
## 10 DAVISON                    I75                   55                    70
## # i 267,068 more rows
```

```
counts <- df %>%
  filter(corrected_speed_limit %in% c(0, 90, 95)) %>%
  group_by(corrected_speed_limit) %>%
  summarise(count = n())

counts
```

```
## # A tibble: 3 x 2
##   corrected_speed_limit count
##                   <dbl> <int>
## 1                     0  4565
## 2                    90     1
## 3                    95     1
```

Then answer: * Were you able to fix all the anomalies with this technique? * What else would you do to fix the remaining erroneous values? - this is an open ended question to help you think through the possibilities. No code necessary to answer this.

After the operation, we still have anomalies for speed limit. For the solution, we can remove this rows from the data frame, or we can use the speed limit with the scondly high frequency for these anomalies.