

CMPD Traffic Stops

Chase Romano

10/07/2019

Analyzing CMPD Traffic Stops

This project examines a data set of stops by the Charlotte-Mecklenburg Police Department (CMPD).

The focus is to understand what factors are related to whether someone is searched or not for a traffic stop.

```
library(tidyverse)
library(scales)
library(ggspatial) # make sure to install if you don't have it

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

Demographics of drivers

First, look at the data using the `glimpse()` function from `dplyr`

```
glimpse(df)

## Observations: 68,488
## Variables: 17
## $ Month_of_Stop      <chr> "2017/06", "2017/06", "2017/06", "201...
## $ Reason_for_Stop    <chr> "Vehicle Regulatory", "Vehicle Regula...
## $ Officer_Race        <chr> "White", "White", "White", "White", "...
## $ Officer_Gender      <chr> "Female", "Male", "Male", "Male", "Ma...
## $ Officer_Years_of_Service <dbl> 3, 2, 5, 10, 2, 3, 2, 5, 25, 10, 25, ...
## $ Driver_Race         <chr> "Black", "Black", "Black", "White", "...
## $ Driver_Ethnicity    <chr> "Non-Hispanic", "Non-Hispanic", "Non-...
## $ Driver_Gender       <chr> "Male", "Male", "Male", "Male", "Fema...
## $ Driver_Age          <dbl> 23, 36, 45, 30, 37, 50, 50, 19, 41, 2...
## $ Was_a_Search_Conducted <chr> "No", "No", "No", "No", "No", "No", "...
## $ Result_of_Stop      <chr> "Verbal Warning", "Verbal Warning", "...
## $ CMPD_Division       <chr> "North Tryon Division", "Central Divi...
## $ ObjectID            <dbl> 3001, 3002, 3003, 3004, 3005, 3006, 3...
## $ CreationDate        <dtm> 2019-02-16 10:01:44, 2019-02-16 10:0...
## $ Creator             <chr> "CharlotteNC", "CharlotteNC", "Charlo...
## $ EditDate            <dtm> 2019-02-16 10:01:44, 2019-02-16 10:0...
## $ Editor              <chr> "CharlotteNC", "CharlotteNC", "Charlo..."
```

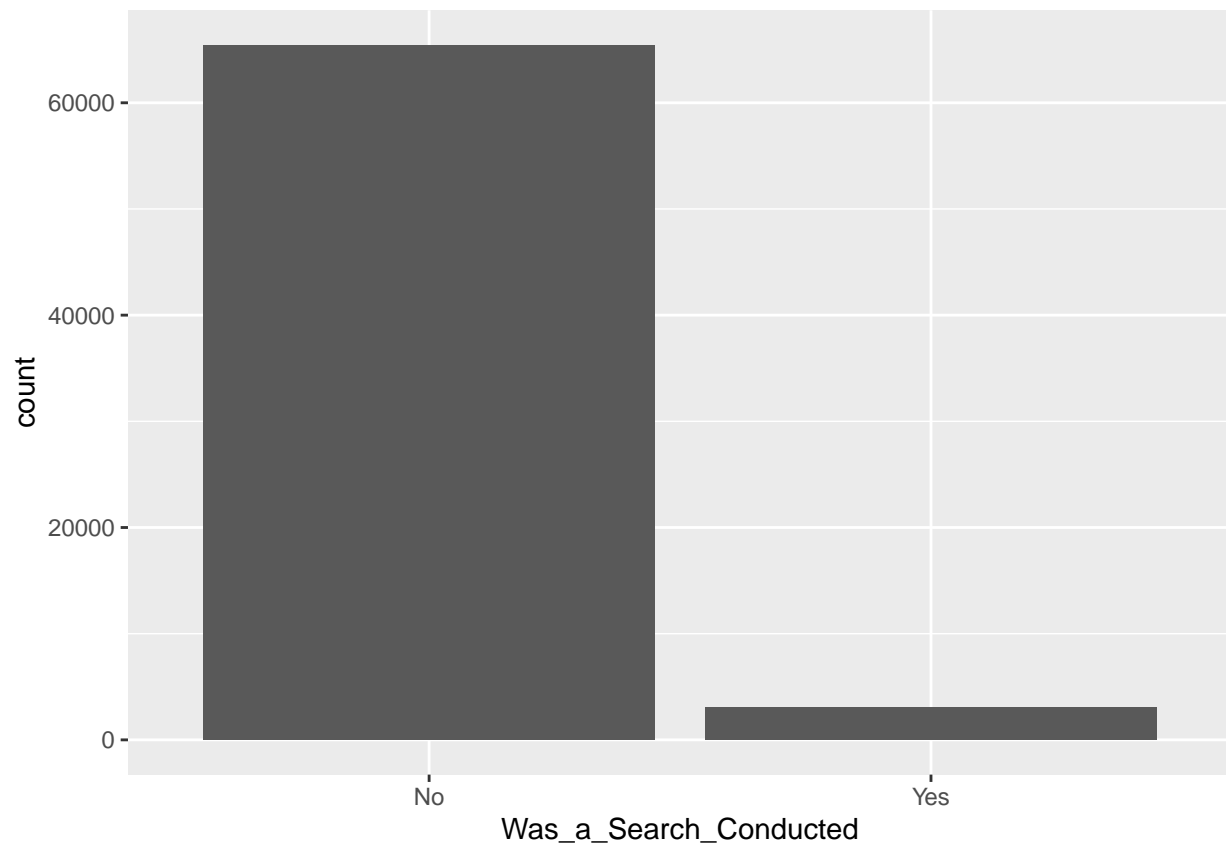
Notice the different variable types: character (`chr`), num (numeric), and datetime (POSIXct).

Let's consider our target variable: `Was_a_Search_Conducted`.

Plot a bar chart that counts the number of records by `Was_a_Search_Conducted`.

```
g <- ggplot(df, aes(Was_a_Search_Conducted)) +
  geom_bar()
```

g

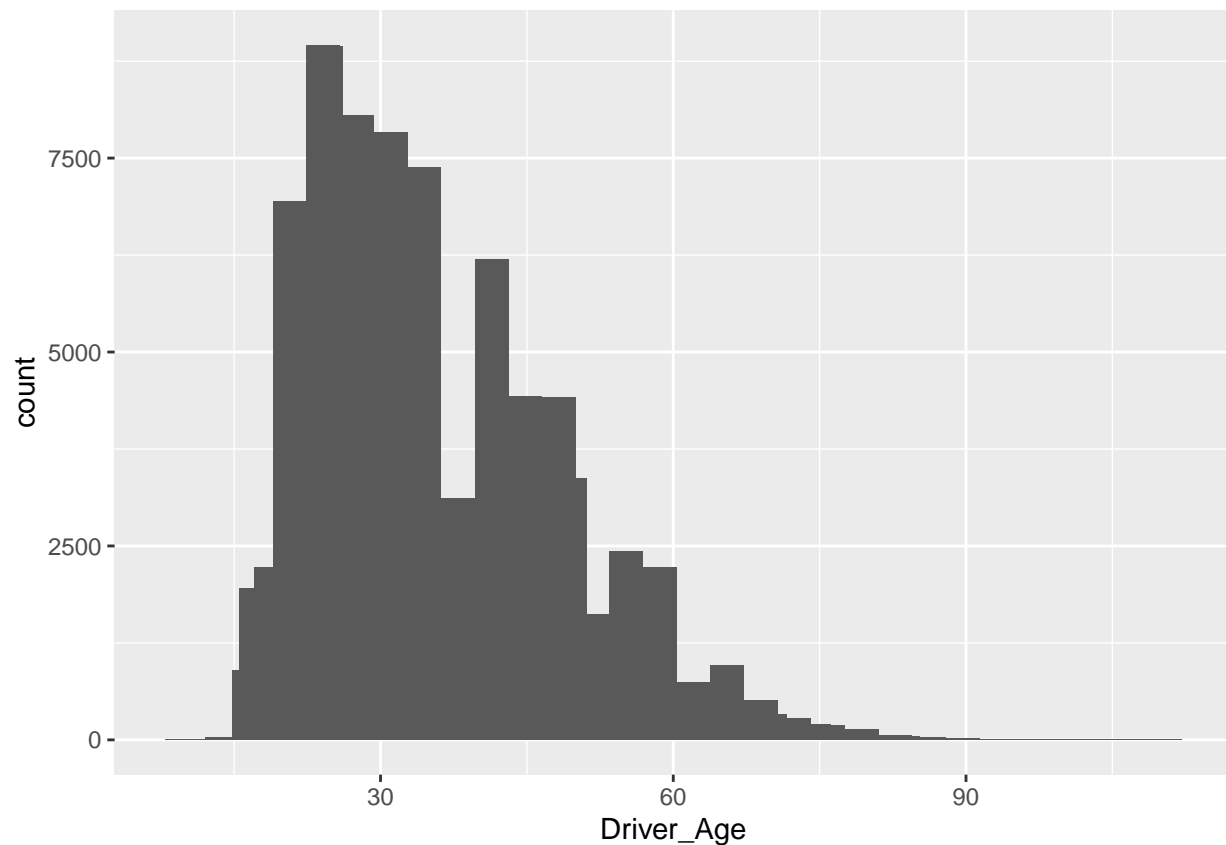


Next, let's consider the age range of the driver.

Plot a histogram of `Driver_Age`. Determine an appropriate number of bins.

```
ggplot(df, aes( x=Driver_Age)) + geom_histogram() + stat_bin(bins = 45)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

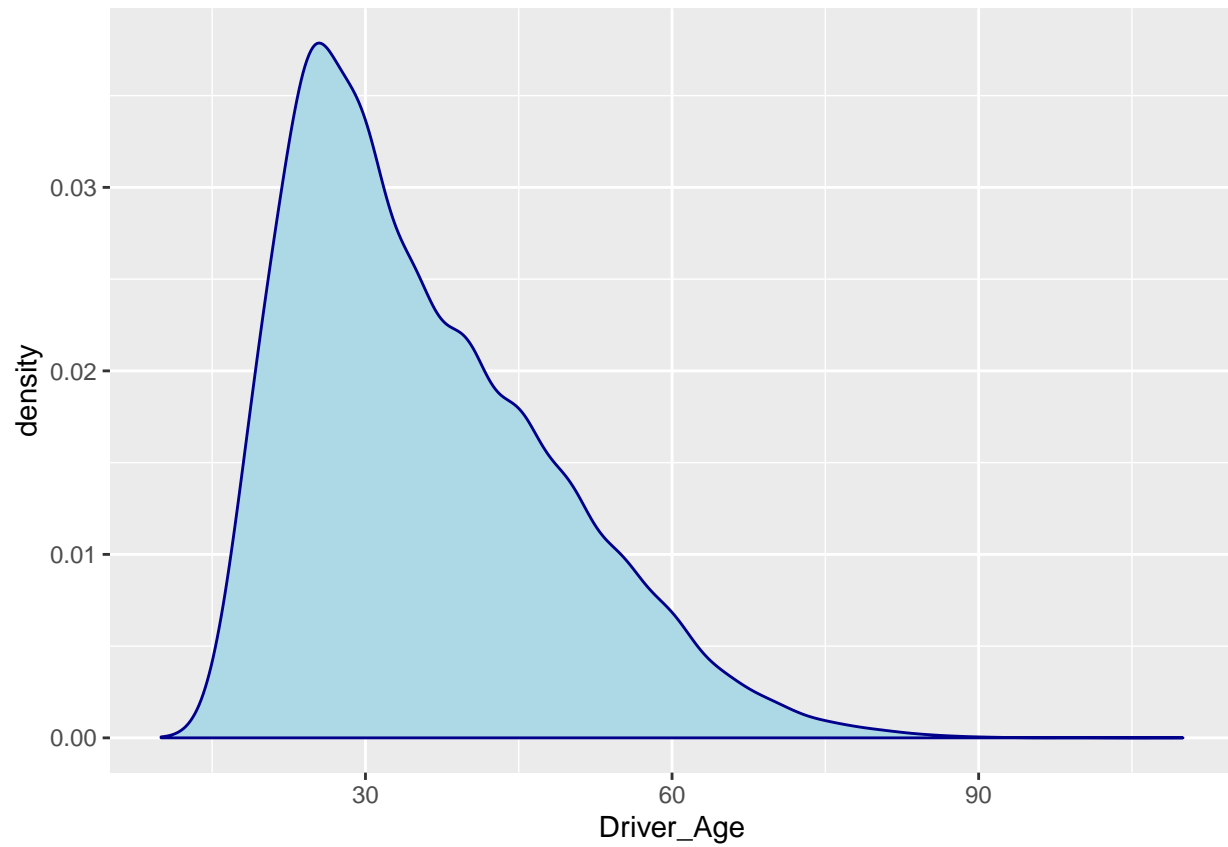


Once you go above (around) 40-50 bins, you'll notice some points stick out.

The data is getting more obviously skewed right

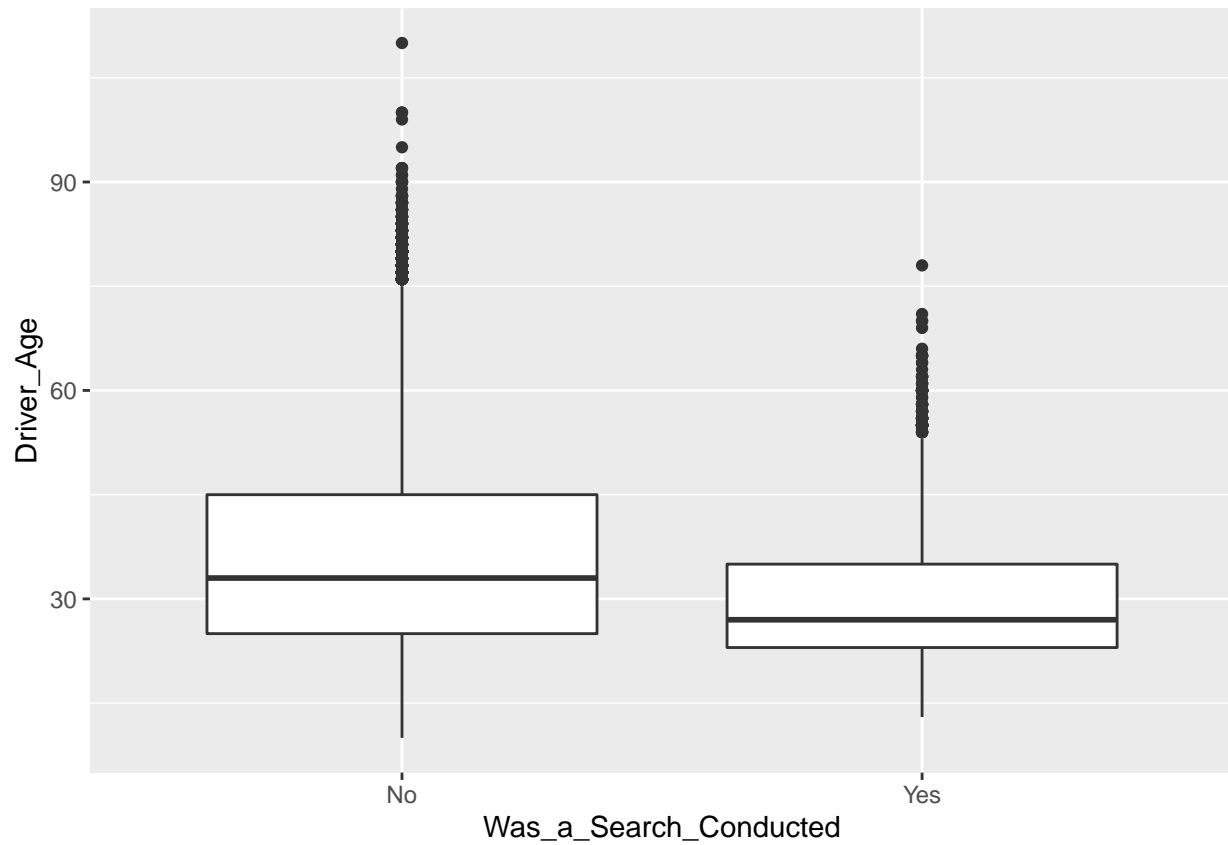
Plot a density plot of `Driver_Age`. Add in a fill to be "lightblue". Determine an appropriate kernel density to use (`adjust`).

```
ggplot(df, aes(x=Driver_Age)) +  
  geom_density(color="darkblue", fill="lightblue", adjust = 1.8)
```



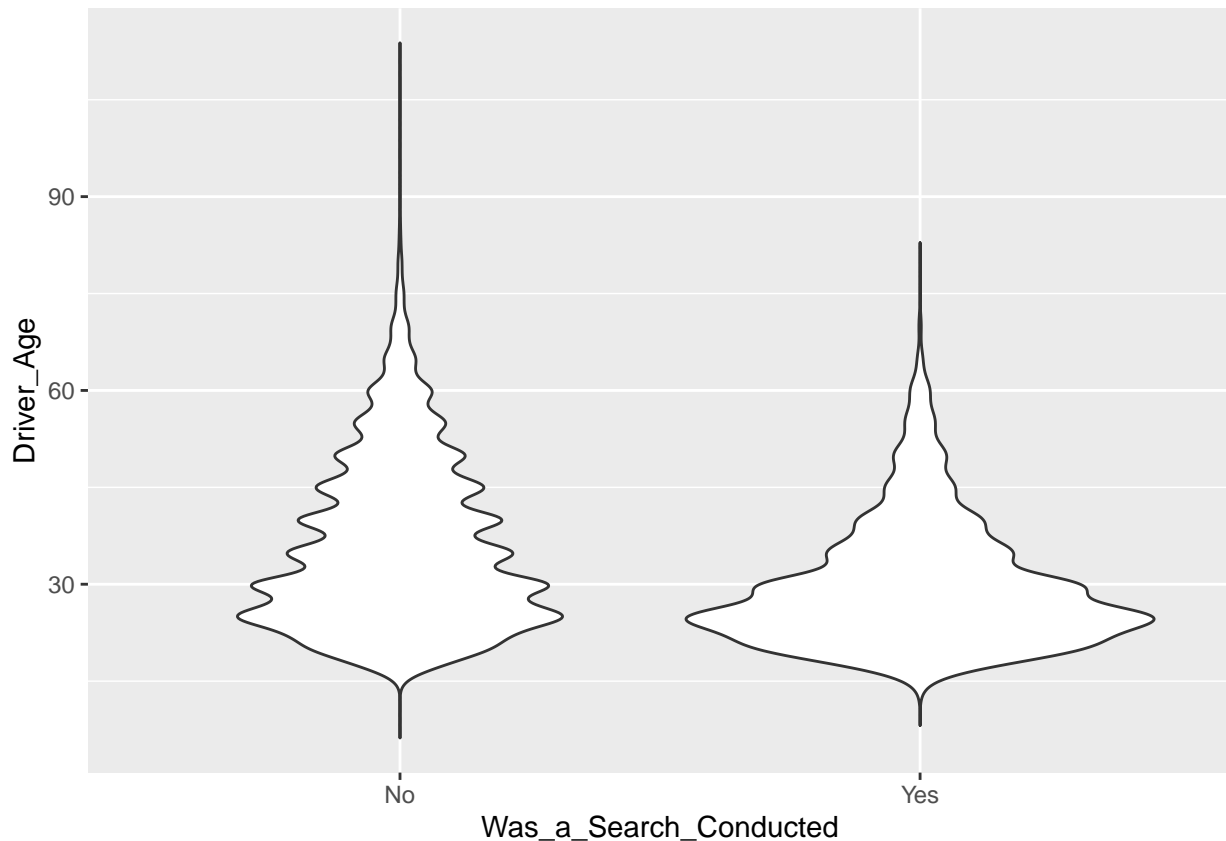
Plot a box plot with Was_a_Search_Conducted on the x-axis and Driver_Age on the y-axis.

```
ggplot(df, aes(x=Was_a_Search_Conducted, y=Driver_Age)) +  
  geom_boxplot()
```



Plot a violin plot.

```
ggplot(df, aes( x=Was_a_Search_Conducted, y=Driver_Age)) + geom_violin(trim=FALSE)
```



From the plots above, do you think the age of the driver is a significant factor in whether a search was conducted? Why or why not?

It seems that more searches are conducted on younger population, there is a larger disparity past 35 years old. Also most of these graphs show the counts so there could also be more of a younger population getting pulled over, causing the data to look like there are getting searched more, I would like to see more of a frequency distribution rather than count

Date of stop

Let's plot the number of stops by time.

Recalling part one, the `Month_of_Stop` variable is a character, not a date variable. The datetime's are simply when the data was collected; not when the stop occurred. Therefore, we'll need to convert the `Month_of_Stop` variable from a character to a Date format.

Let's first cleanup the date field using tidyverse packages like `stringr` and `lubridate`.

```
library(stringr); library(lubridate)
```

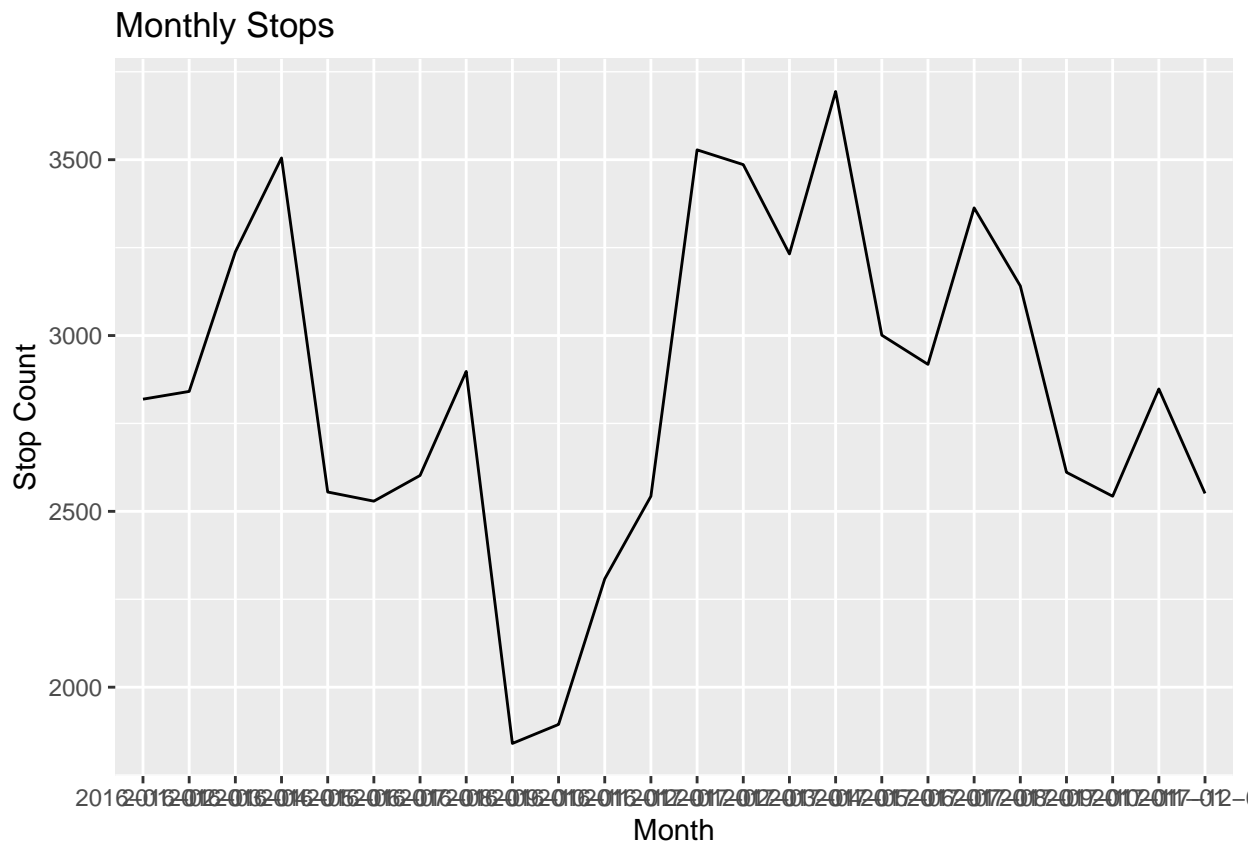
```
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##   date
```

```
# see https://dsba5122fall2019.slack.com/archives/CLUCHHQPJ/p1569273552006200
df <- mutate(df, Month_of_Stop = str_replace_all(Month_of_Stop, "/", "-")) # replace "/" with "-"
```

```
df <- mutate(df, Month_of_Stop = paste0(df$Month_of_Stop, "-01")) # add in day
df <- mutate(df, Date = ymd(Month_of_Stop)) # created a date field
```

Plot a line chart with the number of traffic stops for each month (hint: start with the `count()` function by Date then feed into `ggplot`. Remember the count variable is named 'n').

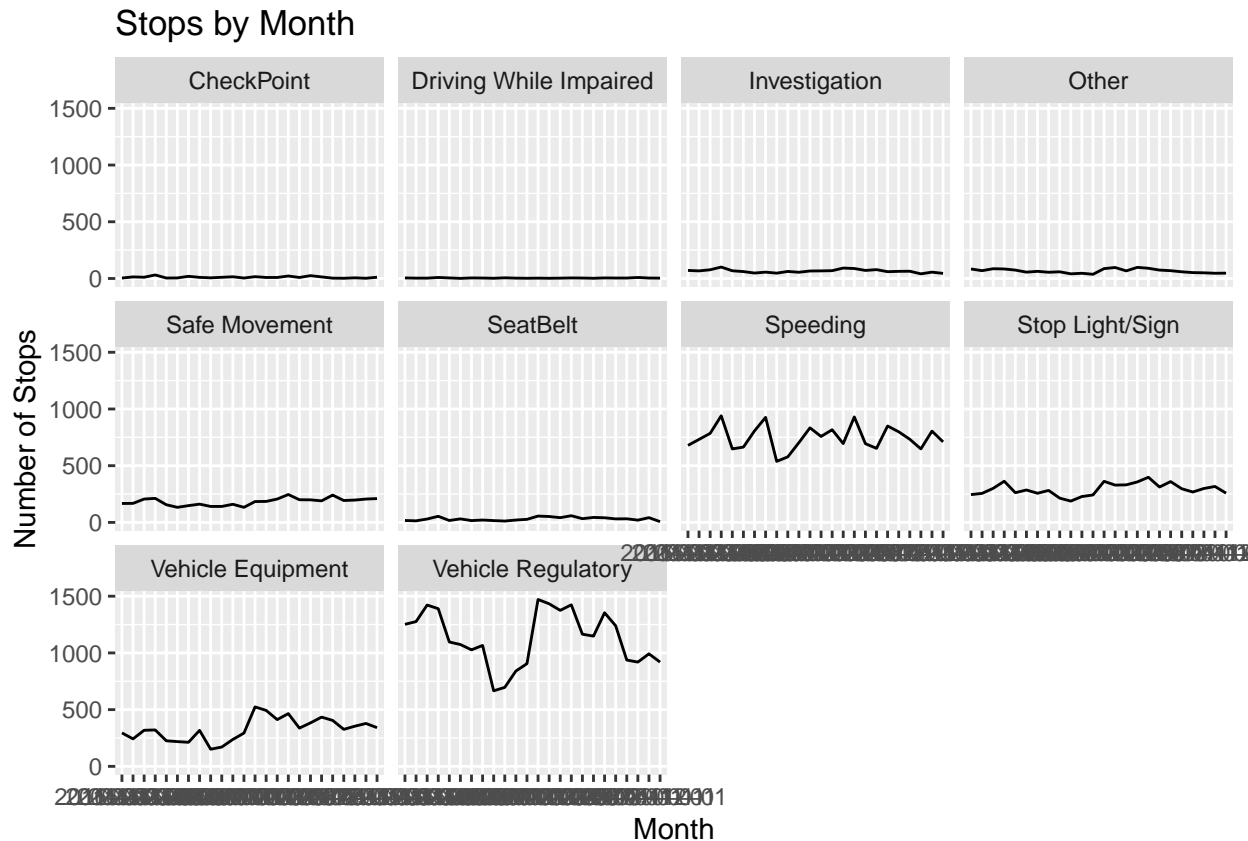
```
df %>%
  count(Month_of_Stop) %>%
  ggplot(aes(x=Month_of_Stop, y= n, group=1)) +
    geom_line()+
    labs(x = "Month", y = "Stop Count",
         title = "Monthly Stops")
```



What is the trend (i.e., long term rate of change) of the number of traffic stops in Charlotte? There seems to be some seasonality but no trend.

Plot the same plot but add in `facet_wrap()` by the `Reason_for_Stop` variable.

```
df %>%
  count(Month_of_Stop, Reason_for_Stop) %>%
  ggplot(aes(x=Month_of_Stop, y= n, group=Reason_for_Stop)) +
    geom_line()+
    labs(x = "Month", y = "Number of Stops",
         title = "Stops by Month")+facet_wrap(~Reason_for_Stop)
```

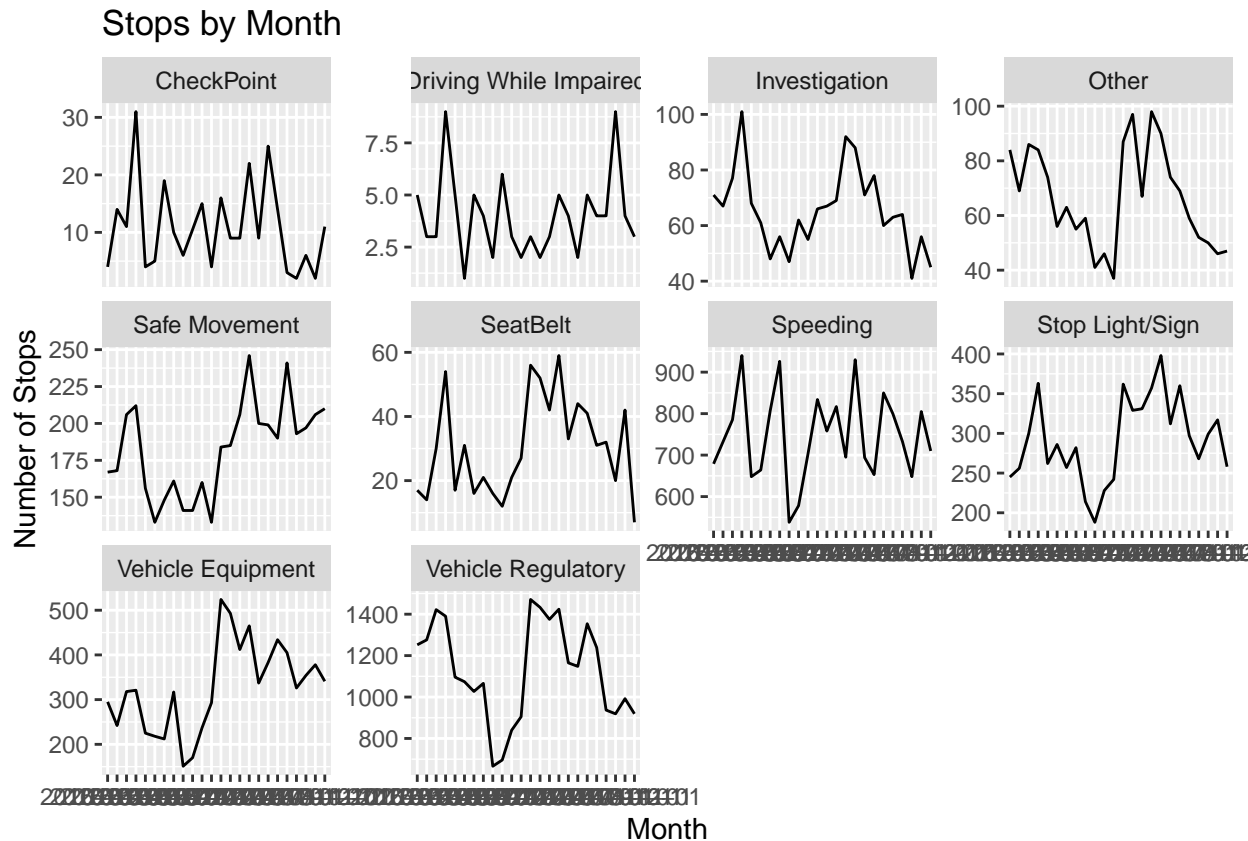


What is a problem with this plot? The scale of the y-axis makes some graphs appear that they are not moving as much as others

To address this problem, you will need to figure out how to adjust the scale. To do this, you need to use R's documentation to see whether there is a parameter in `facet_wrap`.

Plot the same plot but with a free y-axis scale.

```
df %>%
  count(Month_of_Stop, Reason_for_Stop) %>%
  ggplot(aes(x=Month_of_Stop, y= n, group=Reason_for_Stop)) +
    geom_line()+
    labs(x = "Month", y = "Number of Stops",
         title = "Stops by Month")+facet_wrap(~Reason_for_Stop, scales = "free_y")
```

Which type of police stop has had the most volatility (i.e., big swings in number of stops)? Speeding

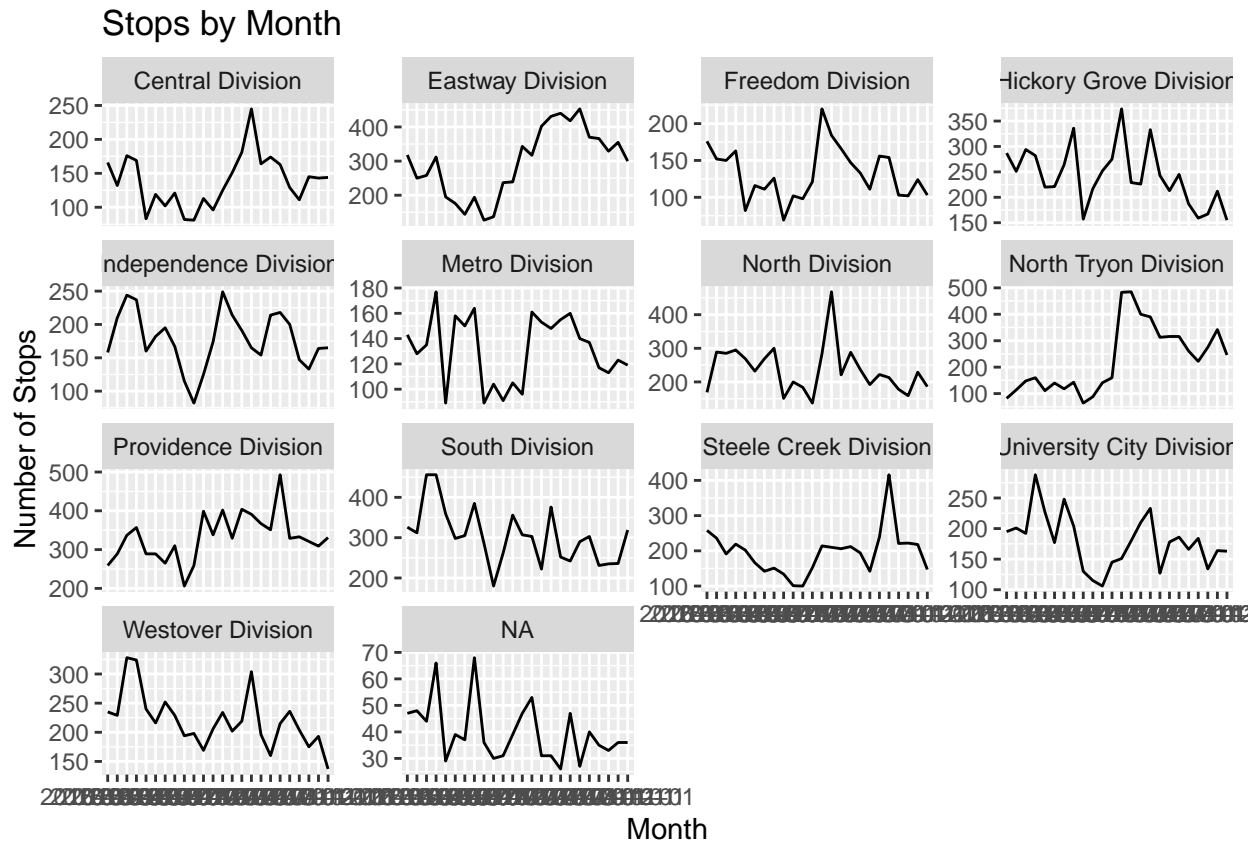
What is one problem with allowing the y-axis be free? The swings may seem larger although the numbers could be much smaller

Small multiples tends to be less effective when each of the variables are on different scales or magnitudes.

Let's consider instead CMPD traffic stops but by CMPD division. These are more even spread by division than the type of stop.

Plot a line chart (optional points too) for stops by Date (x axis) and counts ('n', or whatever you named your count variable) (y axis). (hint: to modify how the date is shown, use the layer `scale_x_date(date_labels = "%Y")` + to show only the year. Feel free to modify by looking at `?scale_x_date.`)

```
lp<-df %>%
  count(Month_of_Stop,CMPD_Division) %>%
  ggplot(aes(x=Month_of_Stop,y= n,group=CMPD_Division)) +
  geom_line()+
  labs(x = "Month", y = "Number of Stops",
       title = "Stops by Month")+facet_wrap(~CMPD_Division,scales = "free_y")
lp
```



What are three observations you can make about the number of police stops by division?

1. Other than one large spike Steele Creek seems to have a constantly low number of stops
2. The Hickory Grove/Metro/South Divisions seem to have a lot of variability
3. Interestingly divisions saw sudden spikes/drops in different months from each other.

Next, this doesn't help tell us where these areas are. For that, let's use a shape file to create a choropleth of stops by division.

Geography

For this example, we'll create a choropleth for the number of police stops by police division.

To do this, we need to use the `sf` package. (For help along the way, see this tutorial on `sf` package.)

```
library(sf); library(viridis)

## Linking to GEOS 3.5.1, GDAL 2.2.2, PROJ 4.9.2
## Loading required package: viridisLite
##
## Attaching package: 'viridis'
## The following object is masked from 'package:scales':
##
##   viridis_pal
```

```
cmpd <- st_read("./data/CMPD_Police_Divisions/CMPD_Police_Divisions.shp")
```

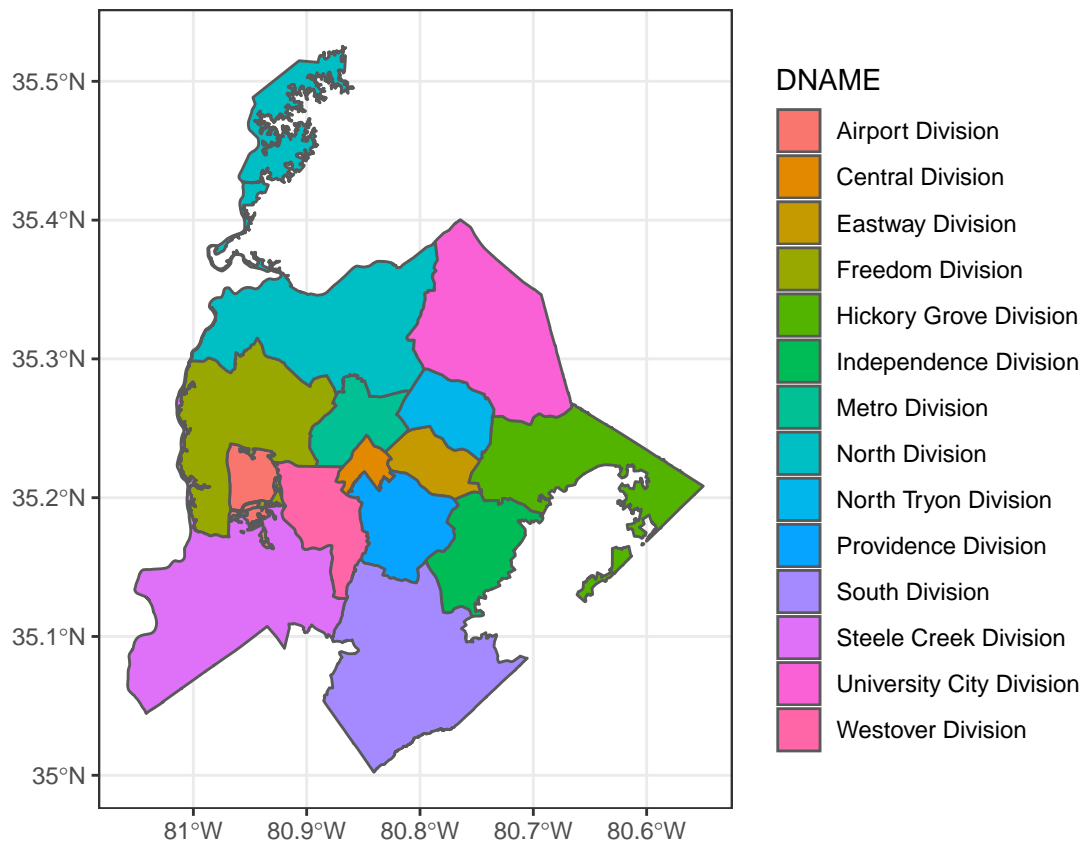
```
## Reading layer `CMPD_Police_Divisions' from data source `/cloud/project/data/CMPD_Police_Divisions/CMPD_Police_Divisions.shp'
## Simple feature collection with 15 features and 12 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -81.05795 ymin: 35.00218 xmax: -80.55006 ymax: 35.52533
## epsg (SRID):    4326
## proj4string:     +proj=longlat +datum=WGS84 +no_defs
```

Note that while we have five files, we only load in the shapefile (.shp) file. This is typical but know that to use this file you would need the other four files in the same folder as your shapefile.

Plot cmpd using the `geom_sf` package where you provide `fill = DNAME` as the only aesthetic. Add in a title saying "CMPD Divisions" and add the `theme_bw()` theme to make translate the file into the black and white template.

```
ggplot(cmpd) +
  geom_sf(aes(fill=DNAME), show.title = "CMPD Divisions")+theme_bw()
```

```
## Warning: Ignoring unknown parameters: show.title
```



```
labs(title = "CMPD Divisions")
```

```
## $title
## [1] "CMPD Divisions"
##
## attr(,"class")
## [1] "labels"
```

One problem with this map is it's hard to read the division names. That is, it may be better to remove the legend and put the labels of each division within the plot.

To do this, we can use the related `geom_sf_label()` geom, using the name of the division as the aesthetic label.

Plot the same plot from above but with the name of the division as the label.

Make sure to remove the legend (it's redundant and no longer necessary).

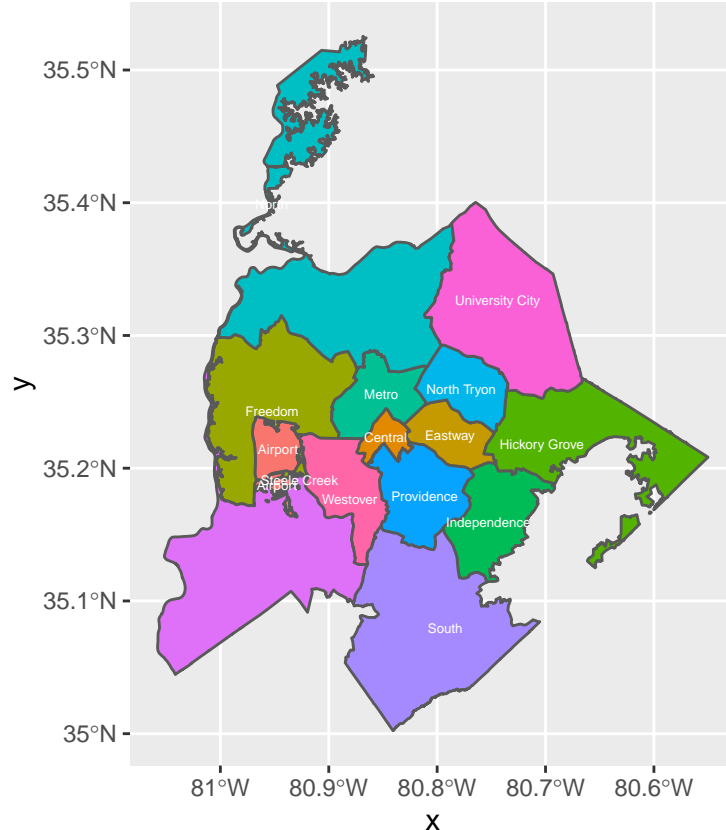
Create a new variable named `Name` that removes the term " Division". This term is redundant and takes up a lot of space in the labels from `DNAME`. To do this step, use this snippet of code at the top of your pipeline:

```
compd %>%  
  mutate(Name = as.character(DNAME)) %>%  
  mutate(Name = str_replace_all(Name, " Division", ""))%>%  
  ggplot() +  
  geom_sf(aes(fill=Name))+ geom_sf_text(aes(label = Name), colour = "white",size=1.8)+ theme(legend.pos=
```

4. Save it as an object named `g`. Make sure to call it once so that the map will output.

```
g<-compd %>%  
  mutate(Name = as.character(DNAME)) %>%  
  mutate(Name = str_replace_all(Name, " Division", ""))%>%  
  ggplot() +  
  geom_sf(aes(fill=Name))+ geom_sf_text(aes(label = Name), colour = "white",size=1.8)+ theme(legend.pos=  
g
```

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



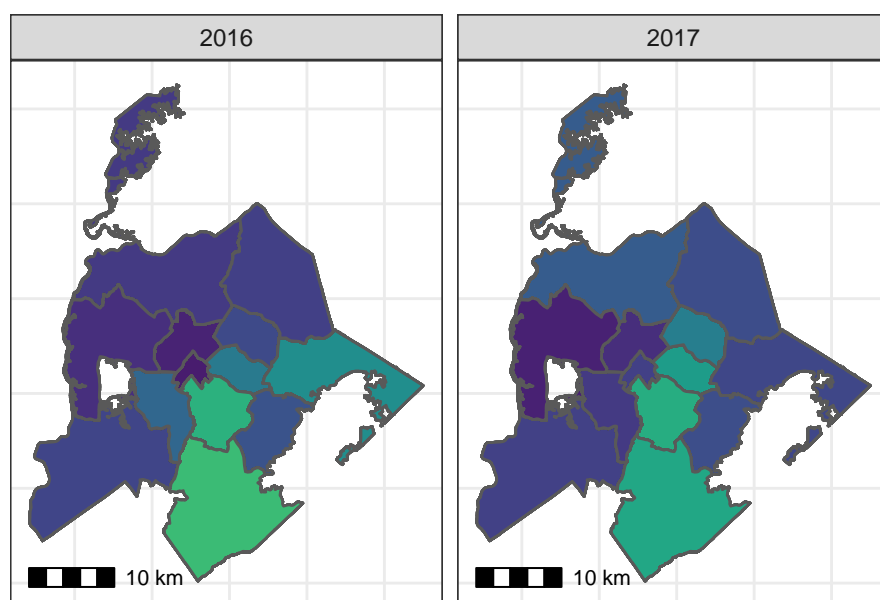
Advanced Plot:

Now, let's create a chloropleth.


```
cmpd_chloropleth <- cmpd %>%  
  mutate(CMPD_Division = as.character(DNAME)) %>%  
  inner_join(count(df, CMPD_Division, Date), by = "CMPD_Division") %>%  
  mutate(Year = lubridate::year(Date)) %>%  
  ggplot() +  
  geom_sf(aes(fill = n)) +  
  scale_fill_viridis("Traffic Stops", labels = scales::comma) +  
  labs(title = "CMPD Traffic stops by CMPD Division",  
       caption = "Source: CMPD") +  
  annotation_scale(location = "bl", width_hint = 0.2) +  
  facet_wrap(~Year) +  
  theme_bw() +  
  theme(legend.position = "bottom",  
        plot.title = element_text(face = "bold", size = rel(1.5)),  
        axis.text.x=element_blank(),  
        axis.text.y=element_blank(),  
        axis.ticks.x=element_blank(),  
        axis.ticks.y=element_blank())
```

cmpd_chloropleth

CMPD Traffic stops by CMPD Division



Traffic Stops



100 200 300 400

Source: CMPD

```
ggsave(cmpd_chloropleth, filename = "cmpd_chloropleth.pdf",  
       width = 7, height = 5, units = "in")  
ggsave(cmpd_chloropleth, filename = "cmpd_chloropleth.png",
```

```
width = 7, height = 5, units = "in")
```

```
##Interactivity
```

For even more fun, **plot** an interactive HTML plot

```
library(plotly)
#
my_cool_plot <- df %>%
  count(Month_of_Stop,CMPD_Division) %>%
  ggplot(aes(x=Month_of_Stop,y= n,group=CMPD_Division)) +
    geom_line()+
    labs(x = "Month", y = "Number of Stops",
         title = "Stops by Month")+facet_wrap(~CMPD_Division,scales = "free_y")
#
my_cool_plot
```

