

# Lab 2

Math 241, Week 2

```
# Put all necessary libraries here
# I got you started!
# The first time you want to install the dsbox package; then you can comment it out.
# If you have not installed the devtools package, you will need to do so first
# install.packages("devtools")
# library(devtools)

devtools::install_github("tidyverse/dsbox")
library(dsbox)
library(tidyverse)
library(viridis)
```

**Due: Thursday, February 8th at 8:30am**

## Goals of this lab

1. Practice coding to adhere to the Tidyverse Style Guide.
2. Practice creating and refining graphs with `ggplot2`.
3. Consider the strengths and weaknesses of various `geoms` and `aesthetics` for telling a data story.

## Notes:

- When creating your graphs, consider context (i.e. axis labels, title, ... )!
- If I provide partially completed code, I will put `eval = FALSE` in the chunk. Make sure to change that to `eval = TRUE` once you have completed the code in the chunk.
- Be prepared to ask for help from me, Simon, and your classmates! We scratched the surface of `ggplot2` in class. But I encourage you to really dig in and make your graphs your own (i.e. don't rely on defaults).

## Problems

### Problem 1: Road traffic injuries in Edinburgh, Scotland

The dataset can be found in the `dsbox` package, and is called `accidents`. It covers all recorded accidents in Edinburgh in 2018; compared to the data set made available by the UK government, some of the variables were modified for the purposes of the package. You can find out more about the data set by inspecting its documentation with `?accidents`. Recreate the following plot, and interpret the results.

```
data("accidents")

accidentsalt <- accidents %>%
  mutate(weekday = case_when(
```

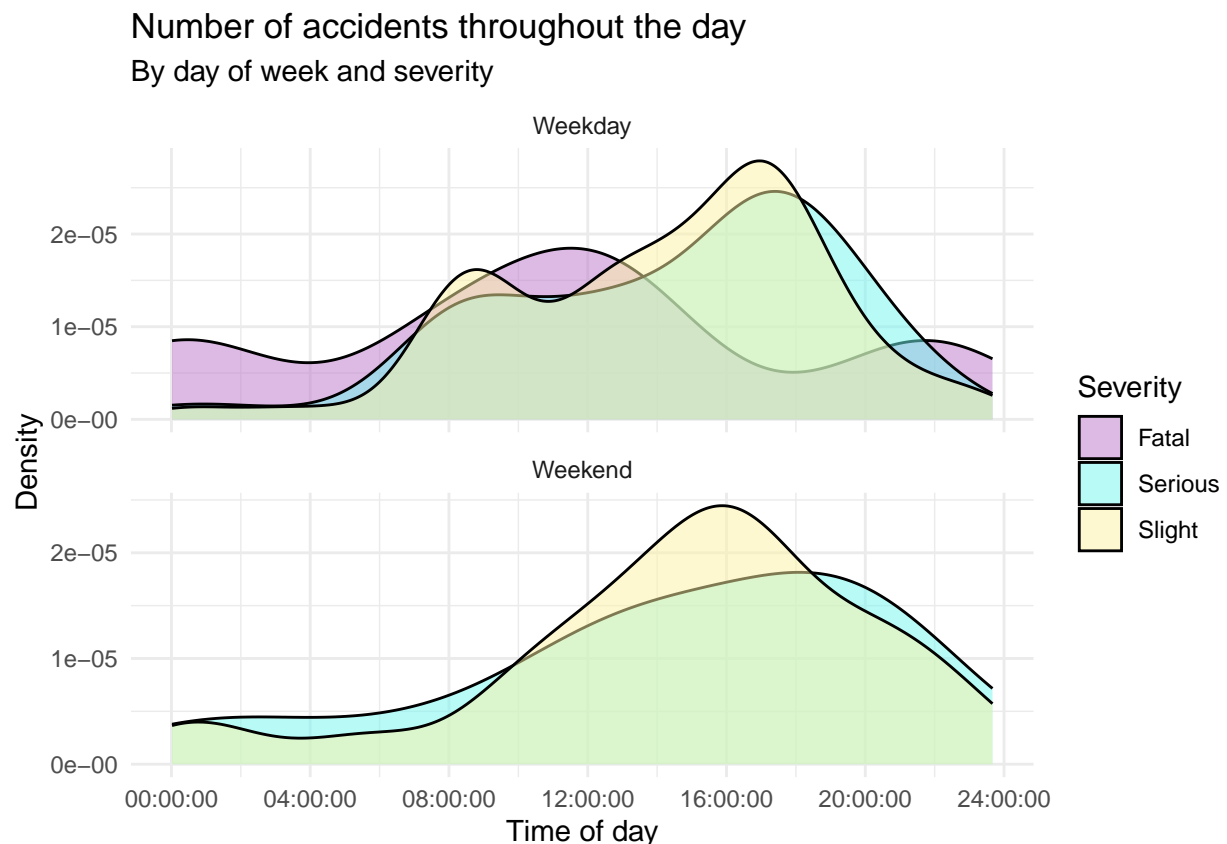
```

  day_of_week %in% c('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday') ~ 'weekday',
  day_of_week %in% c('Saturday', 'Sunday') ~ 'weekend'
))

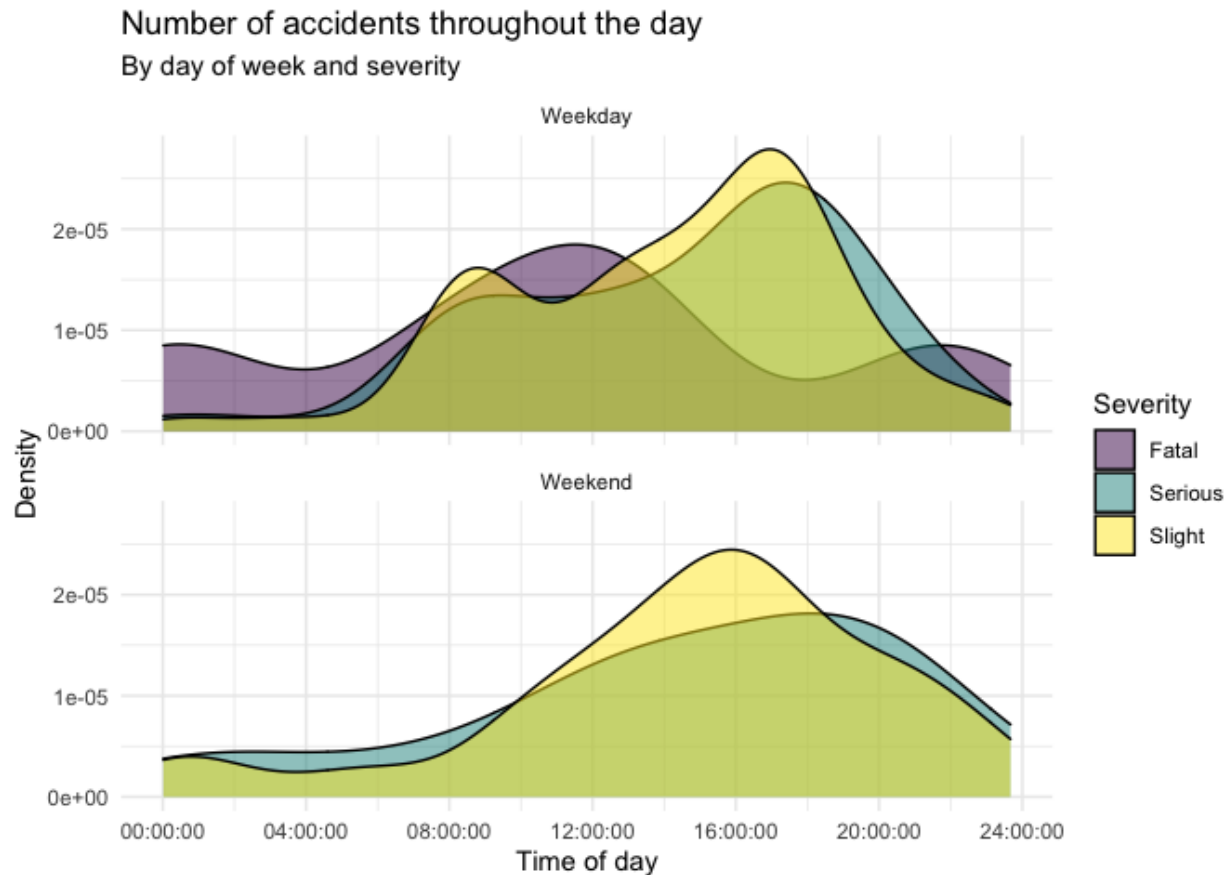
custom_fill_colors <- c("Fatal" = "#bc77cb", "Serious" = "#72f6ef", "Slight" = "#fef3a5")

accidentsalt %>%
  ggplot(aes(x = time)) +
  geom_density(aes(fill = factor(severity)), alpha = 0.5) +
  facet_wrap(~weekday, labeller = labeller(weekday = c(weekday = "Weekday", weekend = "Weekend")), scales = "y",
    scale_y_continuous(breaks = c(0e-00, 1e-05, 2e-05), labels = c("0e-00", "1e-05", "2e-05")) +
  theme_minimal() +
  scale_fill_manual(values = custom_fill_colors) +
  theme(
    strip.text = element_text(hjust = 0.5),
    strip.background = element_blank(),
    strip.placement = "outside") +
  labs(
    title = "Number of accidents throughout the day",
    subtitle = "By day of week and severity",
    x = "Time of day",
    y = "Density",
    fill = "Severity"
  )

```



for this problem I wasn't able to get the exact colors so I went with similar colors but made them fit my personal style a bit more



## Problem 2: One Dataset, Visualized 25 5 Ways

Inspired by Nathan Yau's [One Dataset, Visualized 25 Ways](#), I want you to create 5 visualizations of the same data. You can use the `mpg` data set or another dataset of your choosing, including the `accidents` dataset above. Make sure you have the data manual open for this problem!

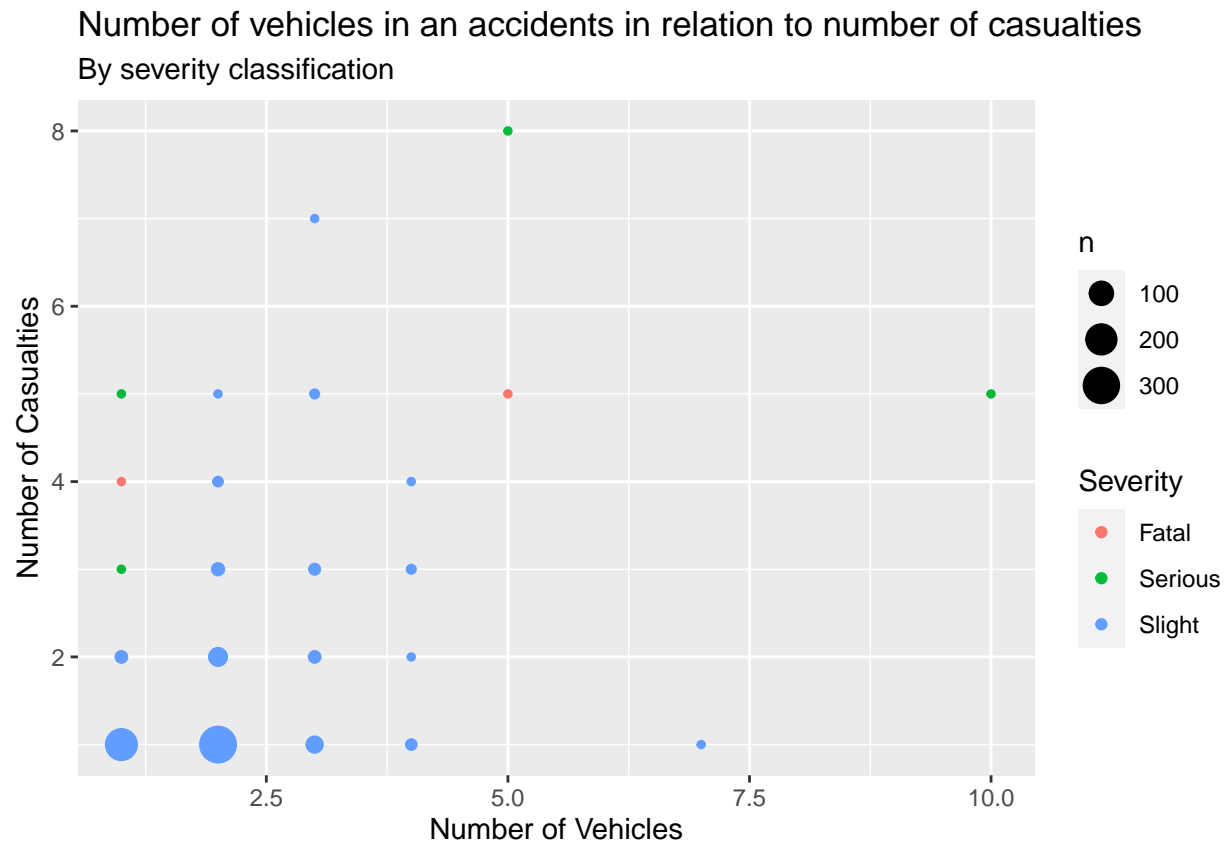
- Pick 3 - 4 variables you want to explore. Provide their code names here. Using `accidents` data set: `vehicles`, `casualties`, `severity`
- Create 5 graphs. A few things to consider:
  - Like Nathan's graphs, they don't all have to contain every one of your selected variables.
  - You can't use the same `geom` for all four graphs but you can use the same `geom` more than once.
  - Think carefully about color, the coordinate system, and scales.
  - Feel free to subset or wrangling the data set if you want to but it isn't required.

```
data("accidents")

accident2 <- accidents

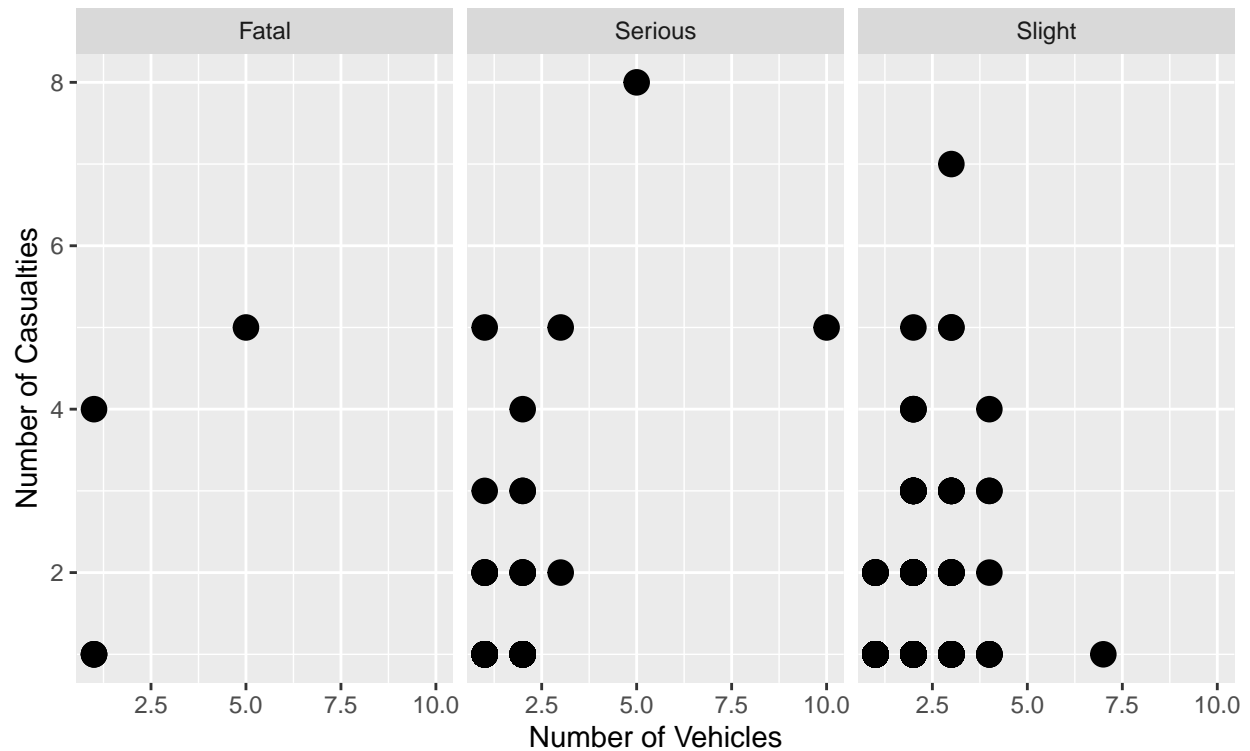
ggplot(data=accident2, aes(x=vehicles, y=casualties, color=severity))+
  geom_count() +
```

```
labs(
  title = "Number of vehicles in an accidents in relation to number of casualties",
  subtitle = "By severity classification",
  x = "Number of Vehicles",
  y = "Number of Casualties",
  color = "Severity"
)
```



```
ggplot(data=accident2, aes(x=vehicles, y=casualties))+
  geom_point(size=4)+
  facet_wrap(~severity) +
  labs(
    title = "Number of vehicles in an accidents in relation to number of casualties",
    subtitle = "By severity classification",
    x = "Number of Vehicles",
    y = "Number of Casualties")
```

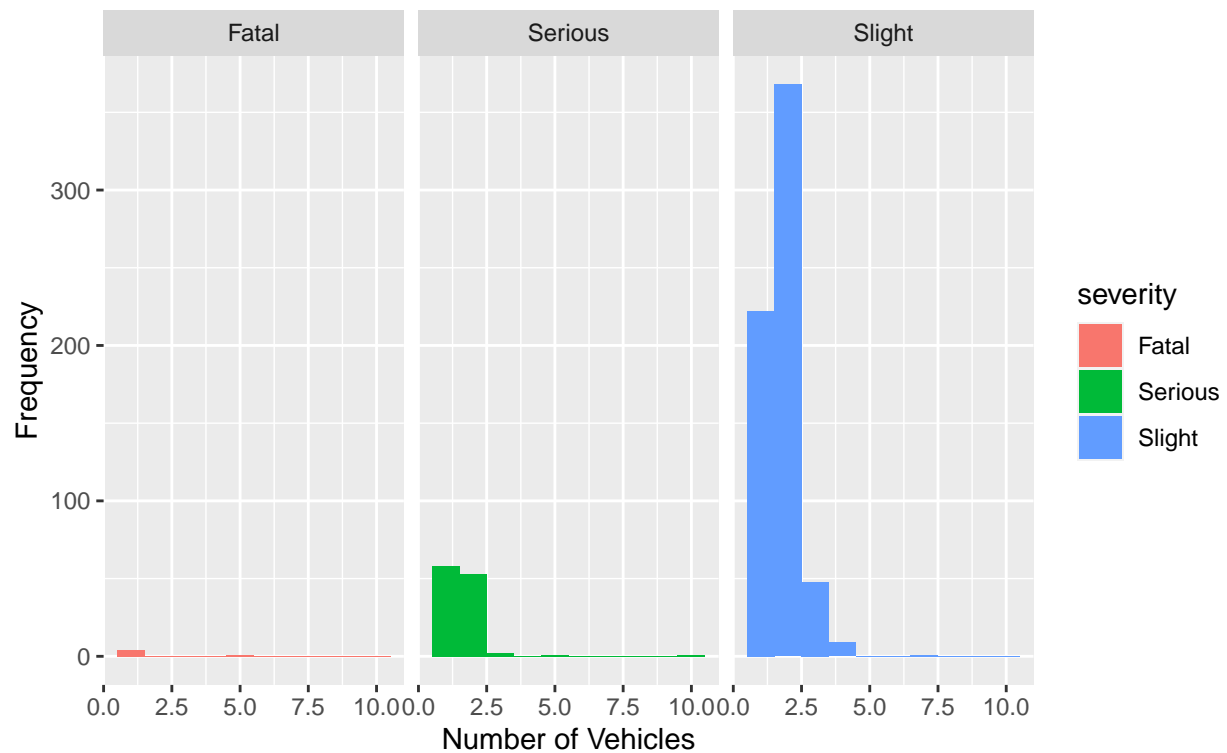
Number of vehicles in an accidents in relation to number of casualties  
By severity classification



```
ggplot(data=accident2, aes(x=vehicles, fill=severity))+
  geom_histogram(binwidth = 1)+
  facet_wrap(~severity) +
  labs(
    title = "Number of vehicles involved in an accident",
    subtitle = "By severity classification",
    x = "Number of Vehicles",
    y = "Frequency",
    color = "Severity")
```

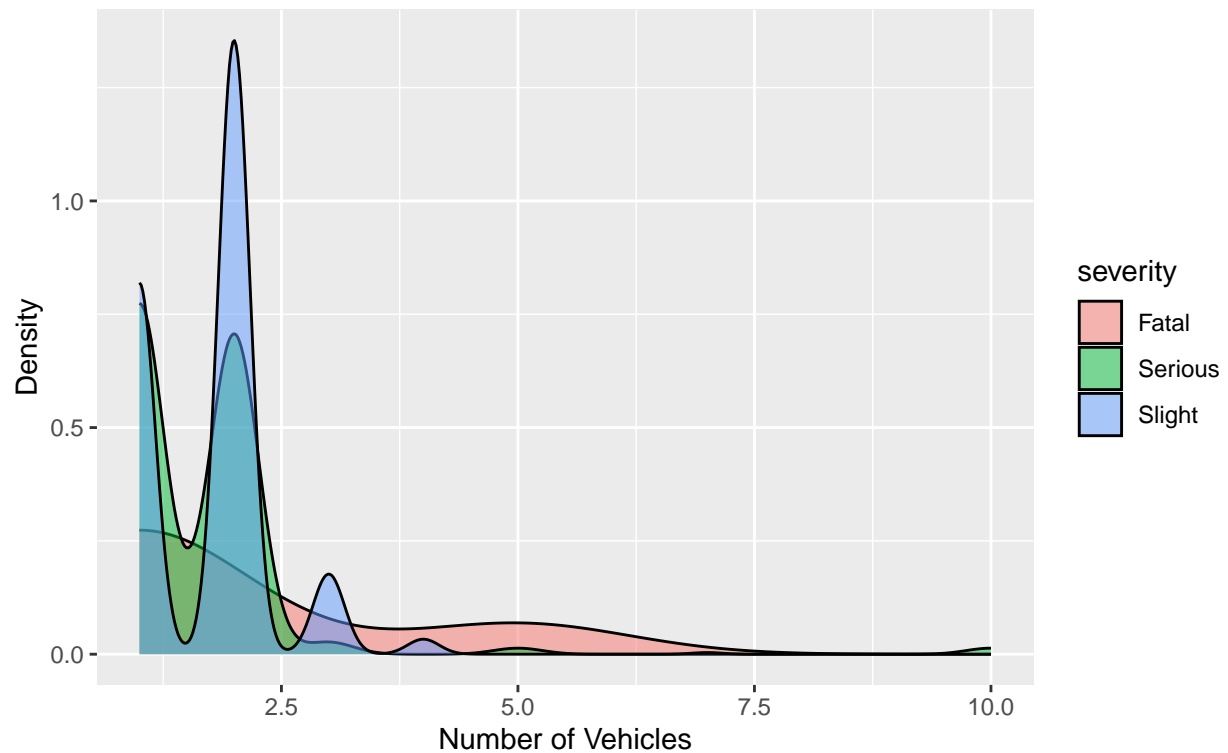
## Number of vehicles involved in an accident

By severity classification

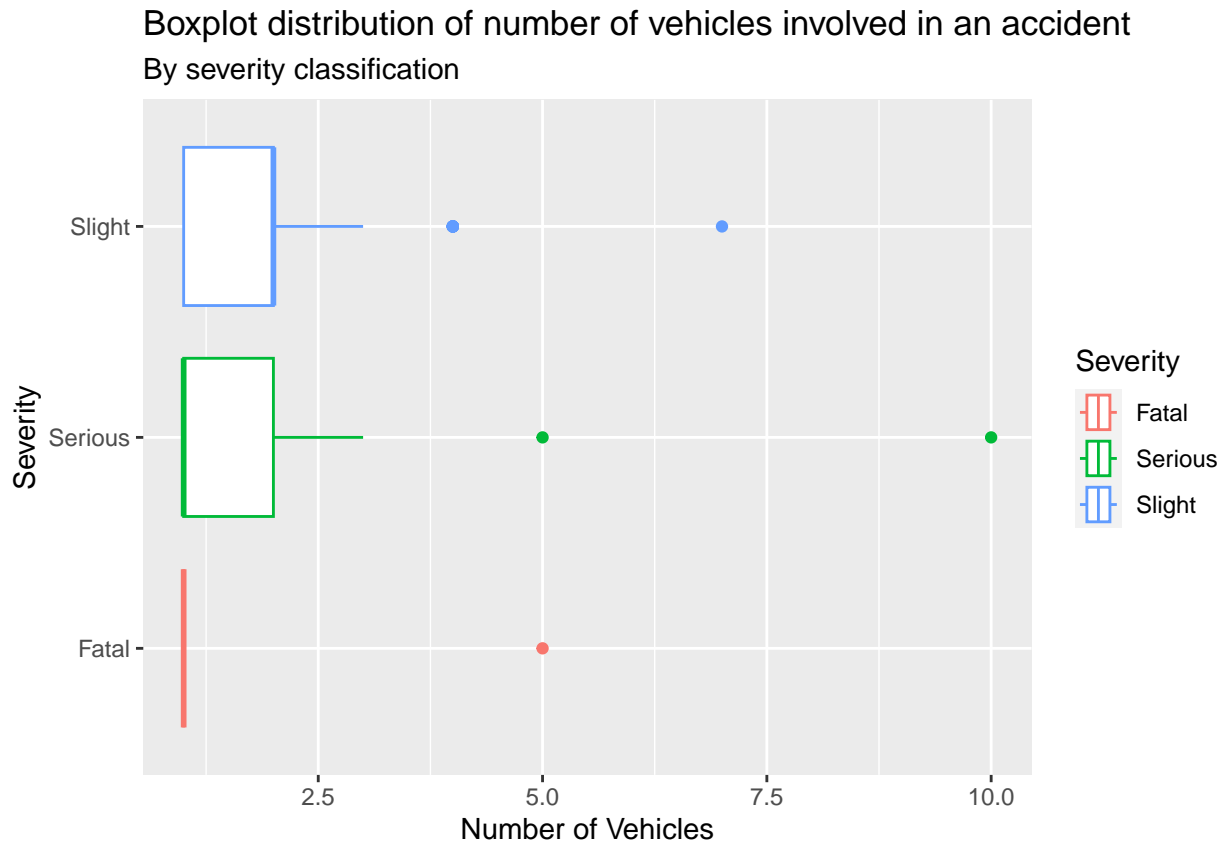


```
ggplot(data=accident2, aes(x=vehicles, fill=severity))+  
  geom_density(alpha=0.5) +  
  labs(  
    title = "Density distribution of number of vehicles involved in an accident",  
    subtitle = "By severity classification",  
    x = "Number of Vehicles",  
    y = "Density",  
    color = "Severity")
```

Density distribution of number of vehicles involved in an accident  
By severity classification



```
ggplot(data=accident2, aes(x=vehicles, y=severity, color=severity))+  
geom_boxplot() +  
  labs(  
    title = "Boxplot distribution of number of vehicles involved in an accident",  
    subtitle = "By severity classification",  
    x = "Number of Vehicles",  
    y = "Severity",  
    color = "Severity")
```



c. Discuss the pros/cons of your graphs. What useful information can be gleaned? How do the different geoms and aesthetics impact the story?

- 1) This graphic looks at all the variables and provides (in my opinion) the clearest and most detailed presentation of data. The only issue which could be fixed with additional work is that smaller points are a little difficult to see.
- 2) The main thing that this graphic does well is show the trends sorted by severity. Unlike the previous which has all the observations mixed, this graphic allows for an easier time at looking at the trend for more specific severity. The main story being told by this graphic is thus the relationship between severity, number of vehicles, and casualties. This graphic is somewhat plain though which in theory could be fixed but I didn't have the time.
- 3) The major drawback with this graphic is that there is a clear concentration of observations within the slight severity category. As a result, it makes observing the trends of the other two categories difficult as they are very small in comparison.
- 4) This tells a similar story to the previous one but it follows a density curve rather than frequency which makes it easier to view all the data for different severity.
- 5) This graphic shows distribution of vehicle counts organized by severity. It is similar to the previous two but it does not provide any concrete numbers like the others (frequency and density values) making it slightly less useful for in depth analysis. However this one unlike the others more clearly shows where the outliers are.



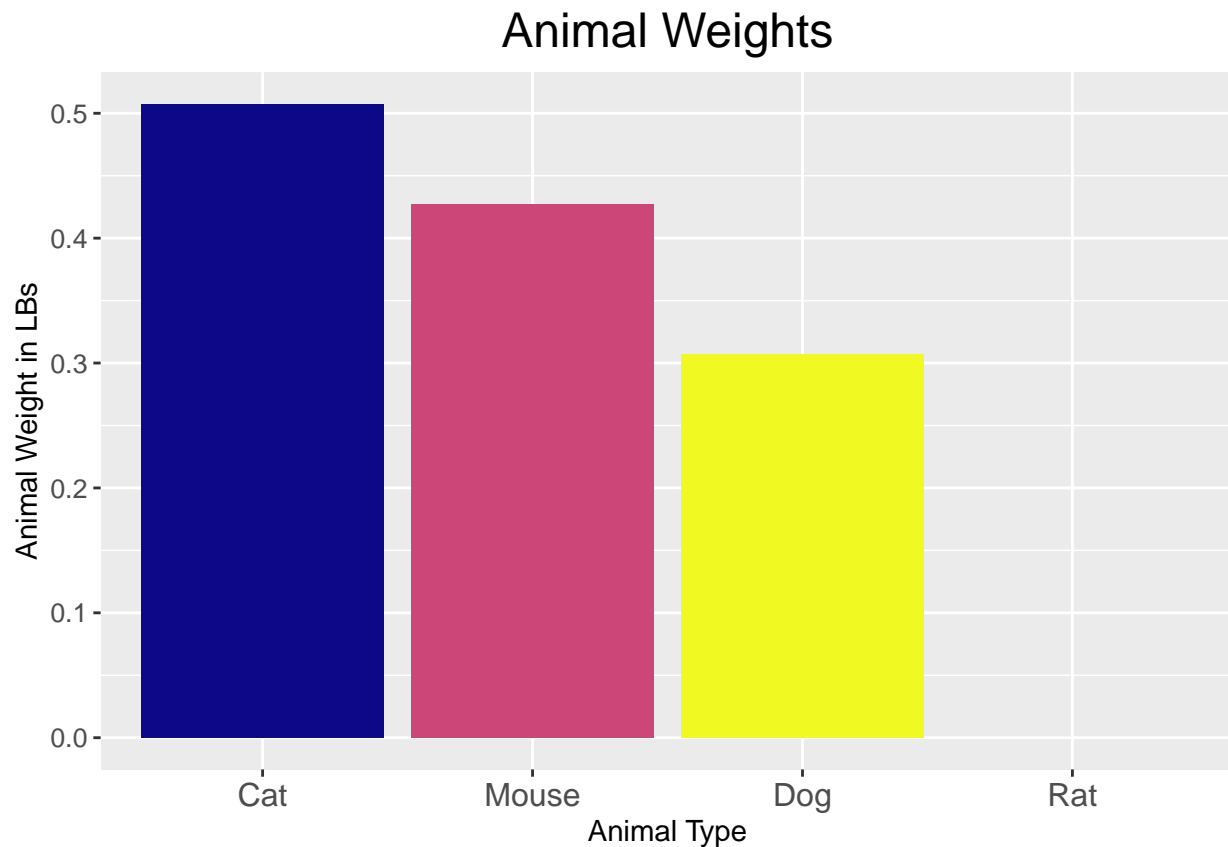
### Problem 3: Style This Code!

Take the following code and don't change its functionality but DO change its style. Use the [Tidyverse Style Guide](#)!

```
set.seed(10)
animals=data.frame(weight=c(runif(3),NA), type=c("cat","mouse","dog","rat"))
median(animals$weight, TRUE);mean(animals$weight, 0 , TRUE); var(animals$weight, NULL, TRUE)

## [1] 0.4269077
## [1] 0.4137181
## [1] 0.01020157

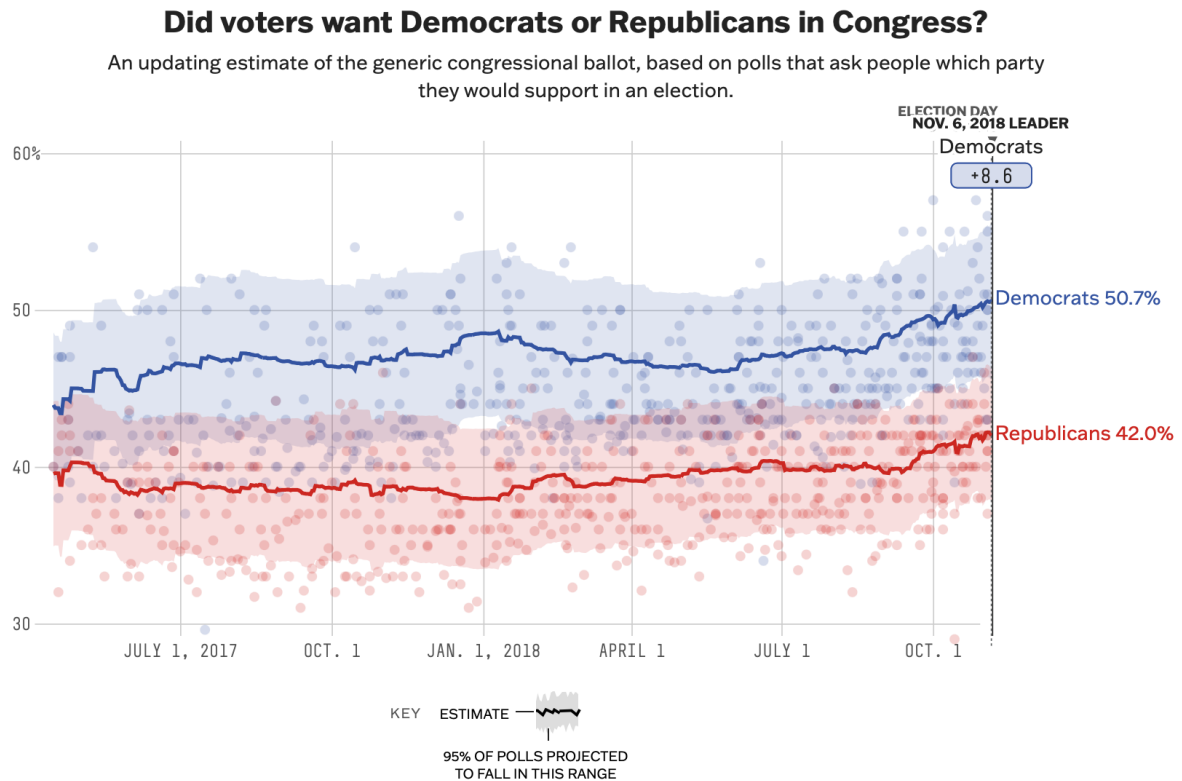
ggplot(animals, aes(x = type, y = weight, fill = type))+
  geom_col()+
  scale_y_continuous()+
  labs(title = "Animal Weights", x = "Animal Type", y = "Animal Weight in LBs", fill = "Animal Type")+
  theme(
    plot.title = element_text(hjust = 0.5, size = 18),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 10),
    legend.position = "none")+
  scale_fill_viridis_d(option = "plasma")+
  scale_x_discrete(labels = c("Cat", "Mouse", "Dog", "Rat"))
```



#### Problem 4: Imitation is the Sincerest Form of Flattery

For this problem, I want you to try to recreate a FiveThirtyEight.com graphic. Awesomely, they share their data with the world [here](#). (Note: You don't need to recreate all their branding/background color scheme.)

- Take a screenshot of the graph, upload it to the same folder on the server where you have saved your lab, and insert the file name below. Then change the `eval = FALSE` to `eval = TRUE`.



- Load the data and recreate the graph as best as you can.

```
library(ggplot2)
library(dplyr)

generic_ballot_averages <- read_csv('generic_ballot_averages.csv')

generic_ballot_averages %>%
  filter(cycle == "2018") %>%
  group_by(candidate) %>%
  ggplot(aes(x = date, y = pct_estimate, ymin = lo, ymax = hi, fill = candidate)) +
  geom_line(size = 1.2) +
  ylim(30, 60) +
  geom_ribbon(alpha = 0.3) +
  scale_fill_manual(values = c("Democrats" = "blue", "Republicans" = "red")) +
  scale_y_continuous(breaks = seq(30, 60, by = 10), labels = paste0(seq(30, 60, by = 10), "%")) +
  scale_x_date(
    breaks = as.Date(c("2017-07-01", "2017-10-01", "2018-01-01", "2018-04-01", "2018-07-01", "2018-10-01"))
```

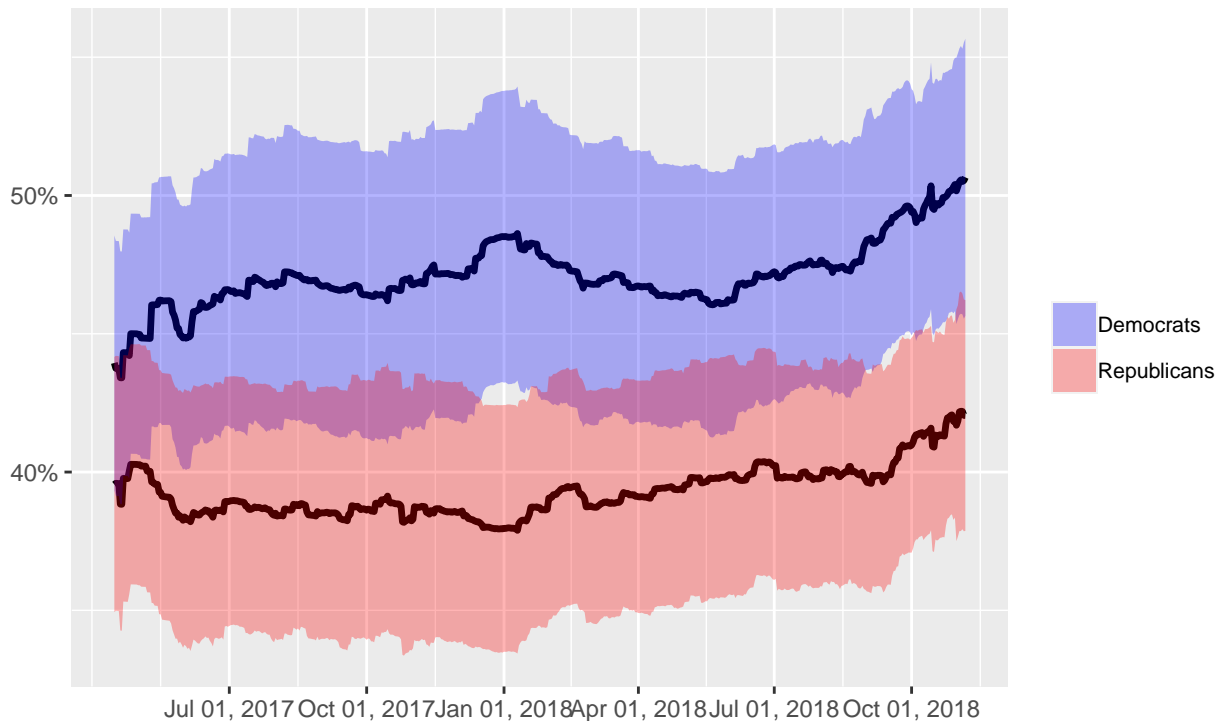
```

labels = c("Jul 1, 2017", "Oct 1, 2017", "Jan 1, 2018", "Apr 1, 2018", "Jul 1, 2018", "Oct 1, 2018")
date_labels = "%b %d, %Y"
) +
labs(
  title = "Did voters want Democrats or Republicans in Congress?",
  subtitle = "An updating estimate of the generic congressional ballot, based on polls that ask people",
  x_axis = " "
) +
theme(
  plot.title = element_text(hjust = 0.5, size = 15),
  plot.subtitle = element_text(hjust = 0.5, size = 10),
  legend.position = "right",
  legend.box.just = "right",
  legend.direction = "vertical",
  legend.text = element_text(size = 8),
  legend.title = element_text(size = 0),
  axis.title = element_blank()
)

```

## Did voters want Democrats or Republicans in Congress?

Updating estimate of the generic congressional ballot, based on polls that ask people which party they would support in an election.



c. Now make the graph better somehow.

See above. Since I was able to make an exact copy some of the small changes that I made have been included in my main graphic,

d. Justify why your rendition of this FiveThirtyEight.com graph is more effective at telling the data story than the original.

The main quality of my graph that I think makes it more effective is that it is much cleaner and easier to read than the original. This is because I was not able to use the points which are on the original graphic but not in the data set. Because of this I think it allows for the reader to more clearly focus on the overall trend modeled by the line and ribbon geoms.

### Problem 5: Rental apartments in SF

The data for this exercise comes from `TidyTuesday`, and is on rental prices in San Francisco. You can find out more about the dataset by inspecting its documentation [here](#). The dataset you'll be using is called `rent`. Create a visualization that will help you compare the distribution of rental prices (`price`) per bedroom (`beds`) across neighborhoods (`nhood`) in the city of San Francisco (`city == "san francisco"`), over time.

Limit your analysis to rentals where the full unit is available, i.e. (`room_in_apt == 0`). You have the flexibility to choose which years and which neighborhoods. Note that you should have a maximum of 8 neighborhoods on your visualization, but one or more of them can be a combination of many (e.g., an “other” category). Your visualization should also display some measure of the variability in your data. You get to decide what type of visualization to create and there is more than one correct answer! In your answer, include a brief description of why you made the choices you made as well as an interpretation of the findings of how rental prices vary over time and neighborhoods in San Francisco.

```
# Get the Data

# Read in with tidyuesdayR package
# Install from CRAN via: install.packages("tidyuesdayR")
# This loads the readme and all the datasets for the week of interest

library(tidyuesdayR)
tuesdata <- tidyuesdayR::tt_load('2022-07-05') # this could take a minute
```

```
##
## Downloading file 1 of 3: 'rent.csv'
## Downloading file 2 of 3: 'sf_permits.csv'
## Downloading file 3 of 3: 'new_construction.csv'
```

```
rent <- tuesdata$rent
```

```
top5_nhood <- unname(unlist(rent %>% filter(city == "san francisco", room_in_apt == 0) %>%
  count(nhood) %>% arrange(desc(n)) %>%
  head(n=5) %>% select(nhood)))

rent_cleaned <- rent %>% filter(city == "san francisco", room_in_apt == 0) %>%
  filter(nhood %in% top5_nhood) %>%
  filter(!is.na(beds))

rent_cleaned %>%
  mutate(beds_clnr = case_when(
    beds %in% c('5', '6', '7', '8') ~ '5+',
    TRUE ~ as.character(beds),
  )) %>%
  ggplot(aes(x = year, y = price, color = nhood)) +
  facet_wrap(~beds_clnr, labeller = labeller(beds_clnr = c('5+' = '5 or more bedrooms', '0' = '0 bedroom
  geom_jitter(alpha=.5)+
  labs(
```

```

title="Rental Prices in the Top 5 San Francisco Neighborhoods Over Time",
subtitle="Organized by Neighborhood and Number of Bedrooms",
x="Year",
y="Price",
color="Neighborhood"
) +
theme(
  plot.title = element_text(hjust = 0.5),
  plot.subtitle = element_text(hjust = 0.5)
) +
scale_fill_brewer(
  type = "qual",
  palette = 7,
  direction = 1,
  aesthetics = "color"
)

```

## Rental Prices in the Top 5 San Francisco Neighborhoods Over Time

Organized by Neighborhood and Number of Bedrooms

