Data Visualization in R with ggplot2

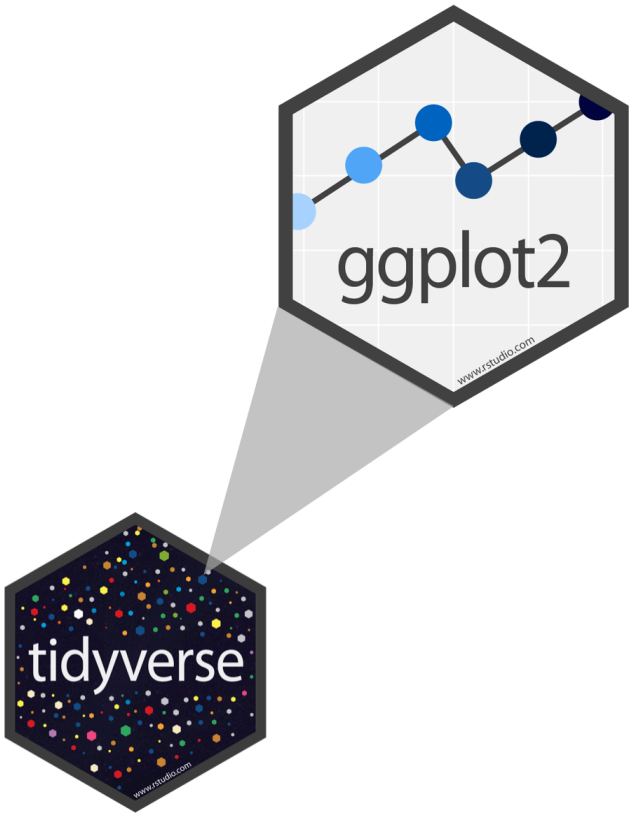
Josh Quan

Introduction

> *"The simple graph has brought more information to the data analyst's mind than any other device."*
> *— John Tukey*

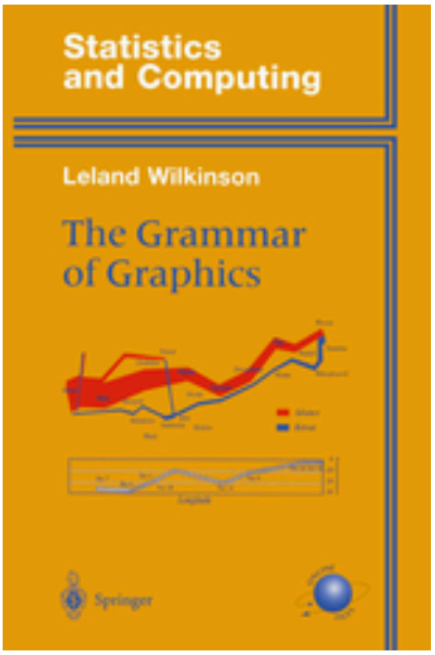
- Data visualization is the creation and study of the visual representation of data.
- Many tools for visualizing data (R is one of them)
- Many approaches/systems within R for making data visualizations, **ggplot2** is one of them

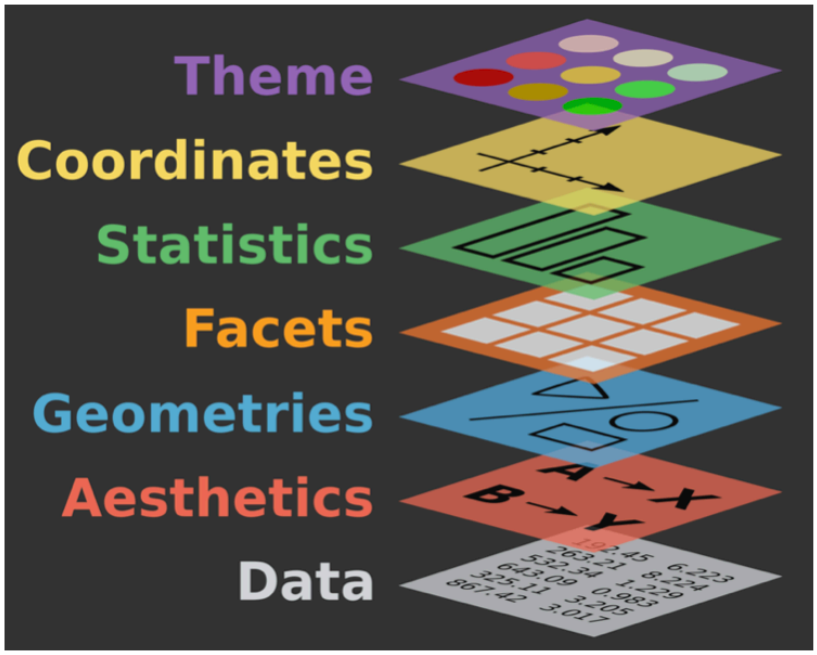
ggplot2 ∈ tidyverse



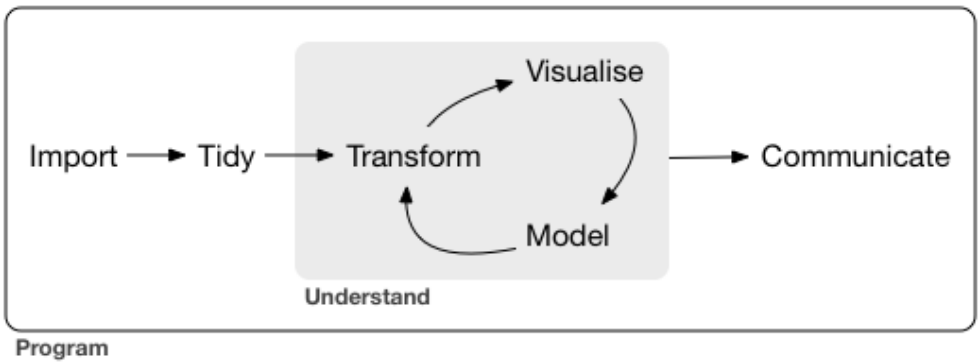
ggplot2 ∈ tidyverse

- **ggplot2**: tidyverse's data visualization package
- gg in "ggplot2" stands for Grammar of Graphics
- Inspired by the book **Grammar of Graphics** by Leland Wilkinson
- A grammar of graphics is a tool that enables concise description of components of a graphic

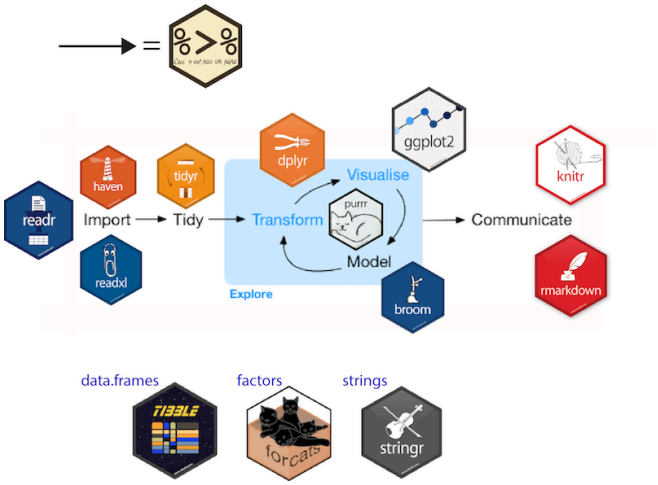

Statistics
and Computing
Leland Wilkinson
The Grammar
of Graphics
Springer


Theme
Coordinates
Statistics
Facets
Geometries
Aesthetics
Data
A → X
B → Y

ggplot2 ∈ tidyverse



ggplot2 ∈ tidyverse



```
library(ggplot2)
library(dplyr)
library(readr)
library(plotly)
library(ggrepel)
library(patchwork)
```

Dataset

Stanford Open Policing Project

- Police Stop Data
  - *state, driver race, stop rate, marijuana legalization status*

```
stops <- readr::read_csv("https://raw.githubusercontent.com/wrathofquan/ggplot2-
        213/refs/heads/main/data/stops.csv")
```

```
str(stops)
```

```
## spc_tbl_ [114 × 7] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ state              : chr [1:114] "CO" "CO" "CO" "CO" ...
##  $ driver_race        : chr [1:114] "white" "white" "white"
"white" ...
##  $ pre_legalization   : logi [1:114] TRUE TRUE TRUE TRUE
TRUE TRUE ...
##  $ quarter            : Date[1:114], format: "2011-02-15"
"2011-05-15" ...
##  $ search_rate        : num [1:114] 0.00454 0.00402 0.00356
0.00373 0.00449 ...
##  $ legalization_status: chr [1:114] "pre" "pre" "pre" "pre"
...
##  $ search_rate_100    : num [1:114] 0.454 0.402 0.356 0.373
0.449 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   state = col_character(),
##   ..   driver_race = col_character(),
##   ..   pre_legalization = col_logical(),
##   ..   quarter = col_date(format = ""),
##   ..   search_rate = col_double(),
##   ..   legalization_status = col_character(),
##   ..   search_rate_100 = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
head(stops)
```

```
## # A tibble: 6 × 7
##   state driver_race pre_legalization quarter    search_rate
legalizati…¹ searc…²
##   <chr> <chr>       <lgl>            <date>           <dbl>
<chr>         <dbl>
## 1 CO    white       TRUE             2011-02-15     0.00454
pre           0.454
## 2 CO    white       TRUE             2011-05-15     0.00402
pre           0.402
## 3 CO    white       TRUE             2011-08-15     0.00356
pre           0.356
## 4 CO    white       TRUE             2011-11-15     0.00373
pre           0.373
## 5 CO    white       TRUE             2012-02-15     0.00449
pre           0.449
```

```
## 6 CO    white       TRUE              2012-05-15     0.00450
pre            0.450
## # … with abbreviated variable names ¹legalization_status, ²
search_rate_100
```

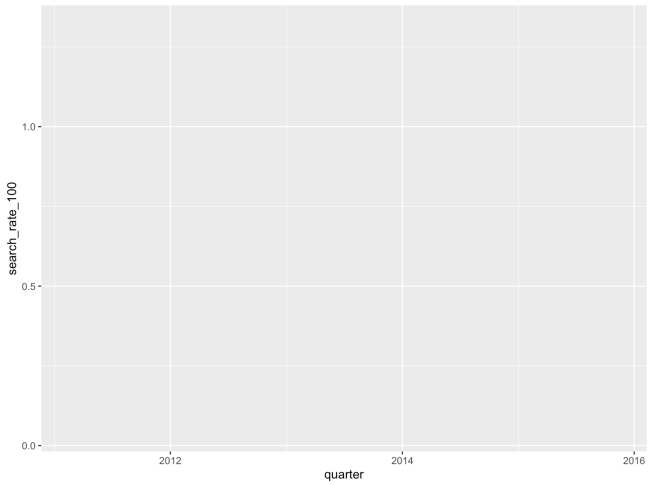
Basic ggplot2 syntax

- Data
- (Aesthetic) mapping to variables
- Geom

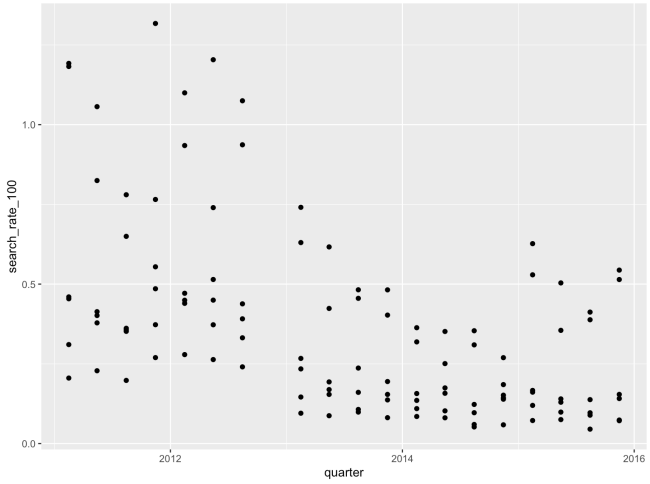
Step-by-step

```
library(ggplot2)
ggplot(data = stops)
```

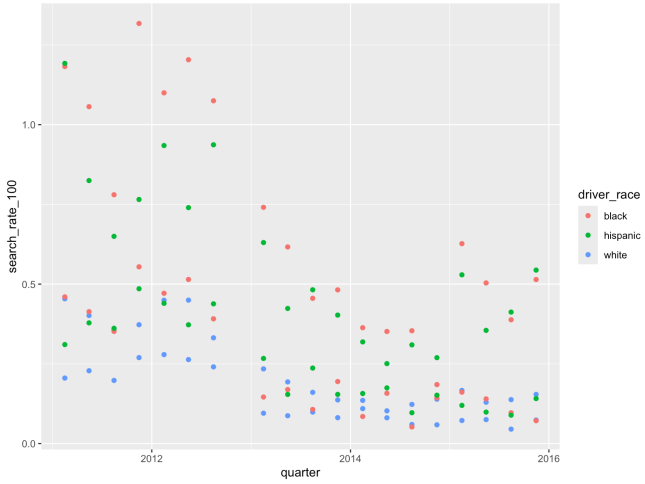
```
ggplot(data = stops, mapping = aes(x = quarter, y = search_rate_100))
```



```
ggplot(data = stops, mapping = aes(x = quarter, y = search_rate_100)) +
  geom_point()
```

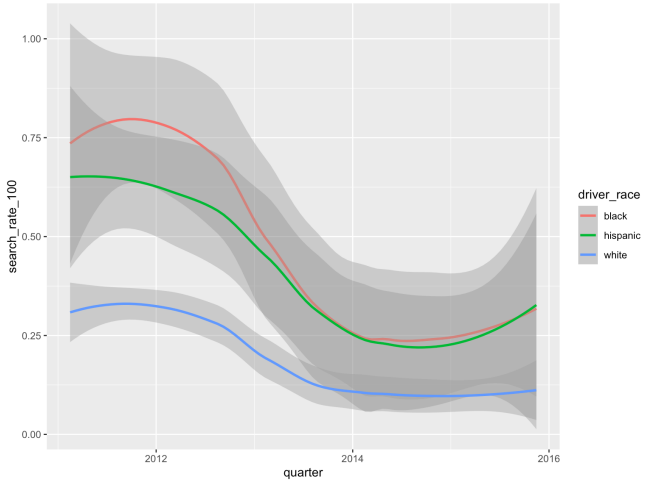


```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point()
```

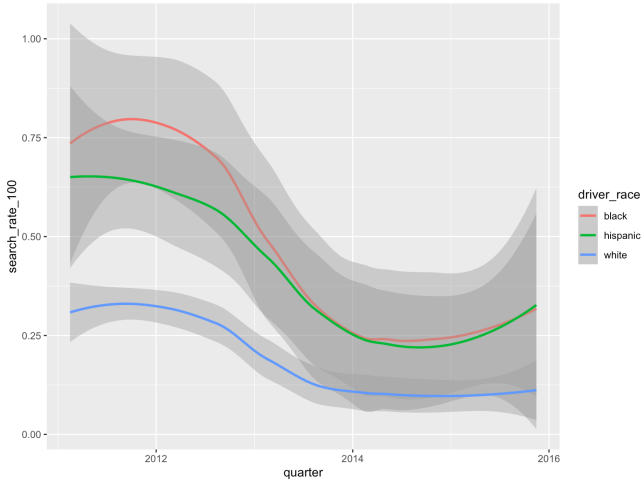


Police Searches Drop Dramatically in States that Legalized Marijuana

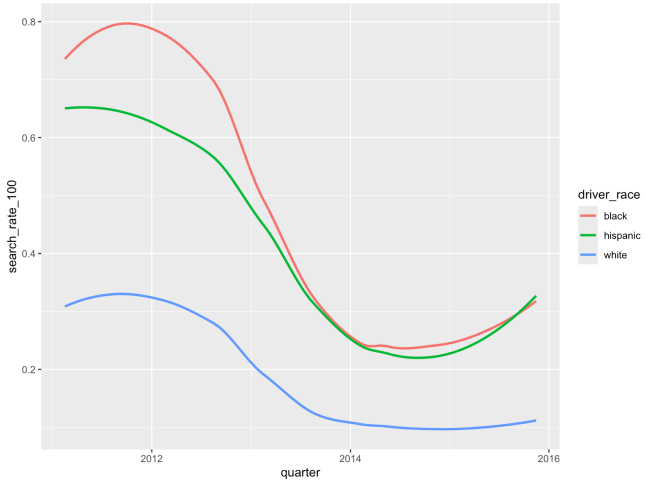
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth()
```



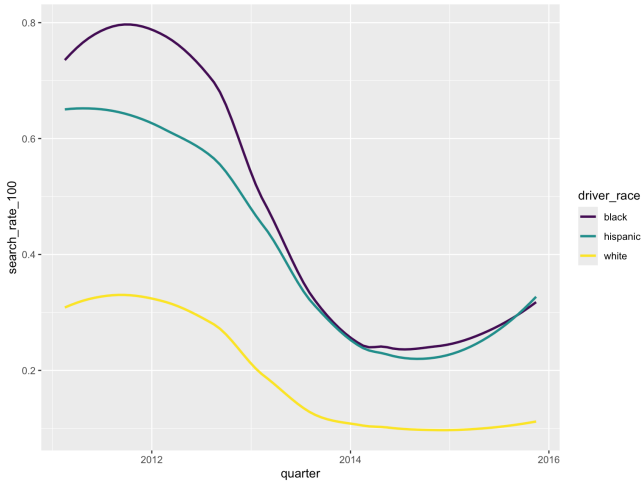
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(method = "loess")
```



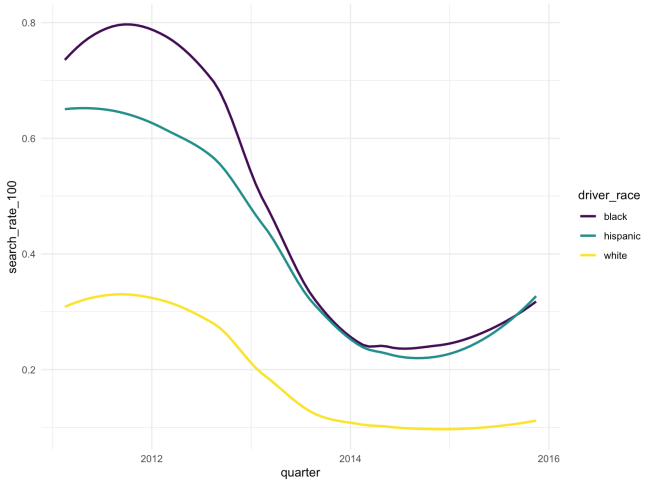
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(method = "loess", se = FALSE)
```



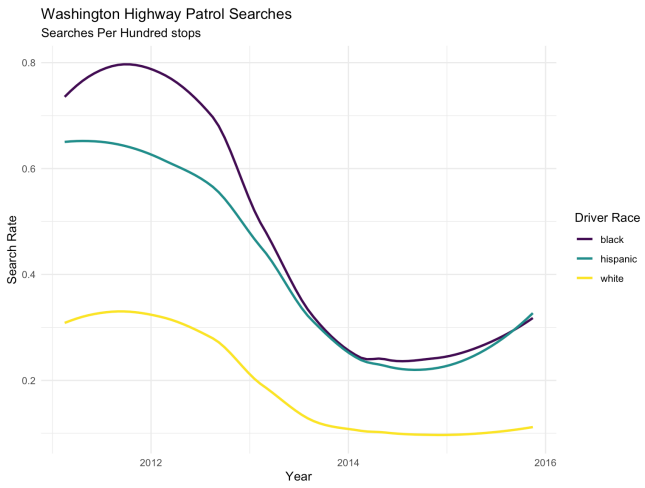
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(method = "loess", se = FALSE) +
  scale_color_viridis_d()
```



```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(method = "loess", se = FALSE) +
  scale_color_viridis_d() +
  theme_minimal()
```



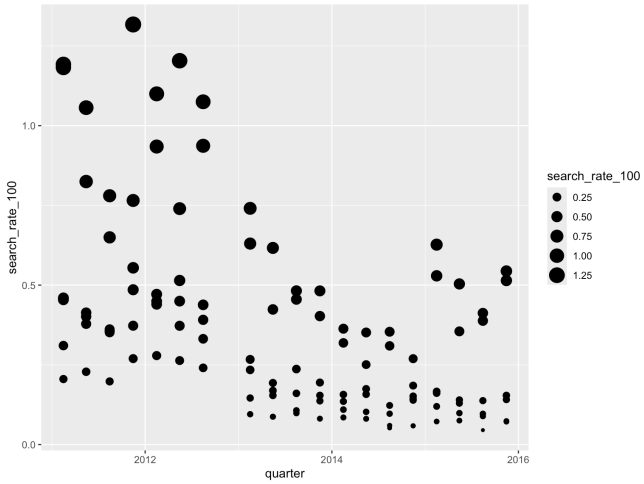
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(method = "loess", se = FALSE) +
  scale_color_viridis_d() +
  theme_minimal() +
  labs(x = "Year", y = "Search Rate", color = "Driver Race",
       title = "Washington Highway Patrol Searches", subtitle = "Searches Per
        Hundred stops")
```



Mapping variables

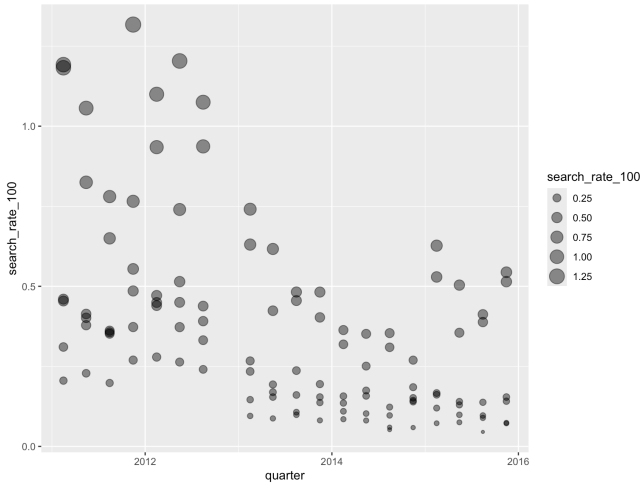
Size data points by a numerical variable

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, size = search_rate_100))
        +
  geom_point()
```



Set alpha value

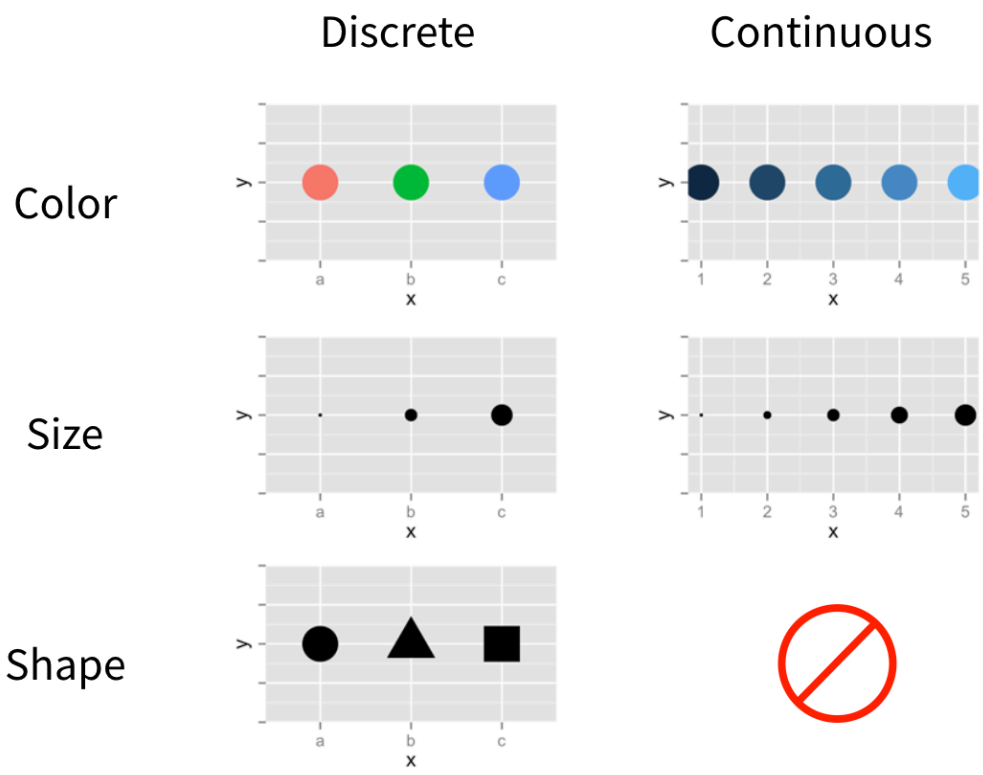
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, size = search_rate_100))
        +
  geom_point(alpha = 0.5)
```



Your turn!

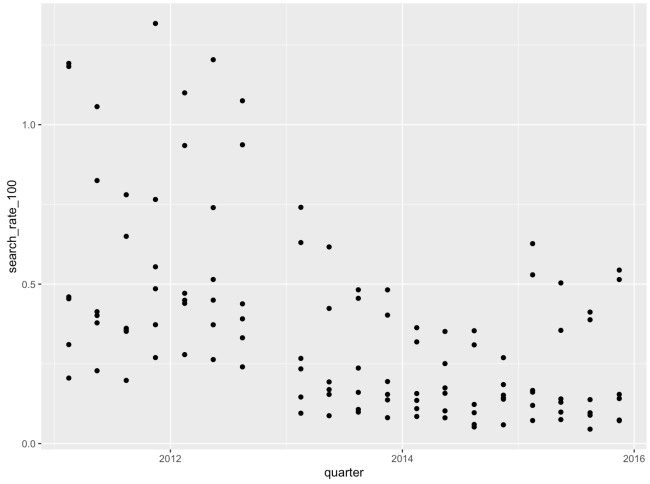
Exercise: Using information from https://ggplot2.tidyverse.org/articles/ggplot2-specs.html add color, size, alpha, and shape aesthetics to your graph. Experiment. Do different things happen when you map aesthetics to discrete and continuous variables? What happens when you use more than one aesthetic?

```
stops %>% ggplot(aes(x = quarter , y = search_rate_100, color = driver_race)) +
  geom_point() +
  theme_minimal(base_size = 12)
```


Discrete
Continuous
Color
Size
Shape
y
x
a
b
c
1
2
3
4
5

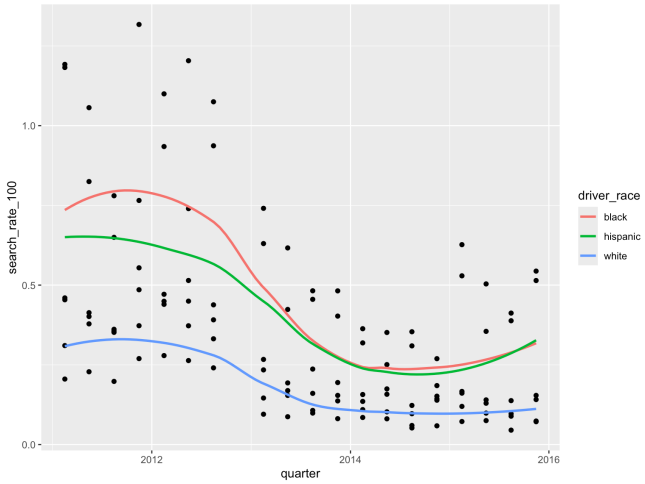
Mappings can be at the geom level

```
ggplot(data = stops) +
  geom_point(mapping = aes(x = quarter, y = search_rate_100))
```



Different mappings for different geoms

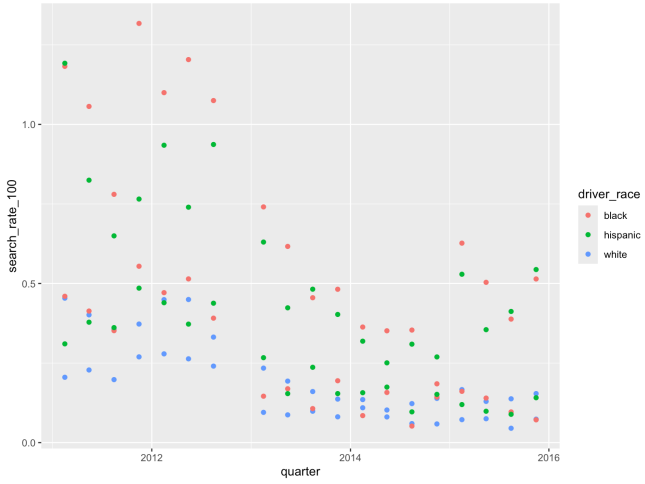
```
ggplot(data = stops, mapping = aes(x = quarter, y = search_rate_100)) +
  geom_point() +
  geom_smooth(aes(color = driver_race), method = "loess", se = FALSE)
```



Set vs. map

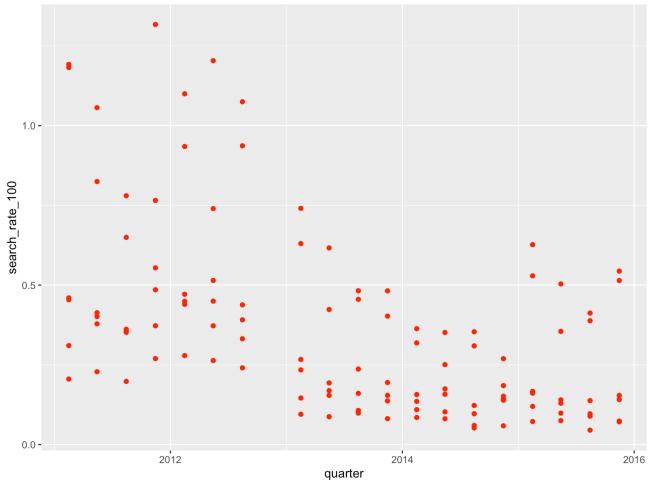
- To **map** an aesthetic to a variable, place it inside `aes()`. Think of this as a *global* rule

```
ggplot(data = stops,
  mapping = aes(x = quarter,
                y = search_rate_100,
            color = driver_race)) +
  geom_point()
```



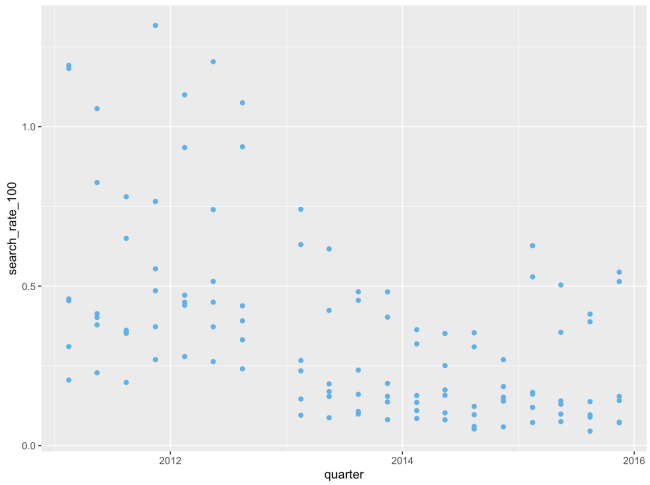
- To **set** an aesthetic to a value, place it outside `aes()`. Think of this as a *local* setting

```
ggplot(data = stops,
  mapping = aes(x = quarter,
                y = search_rate_100)) +
  geom_point(color = "red")
```



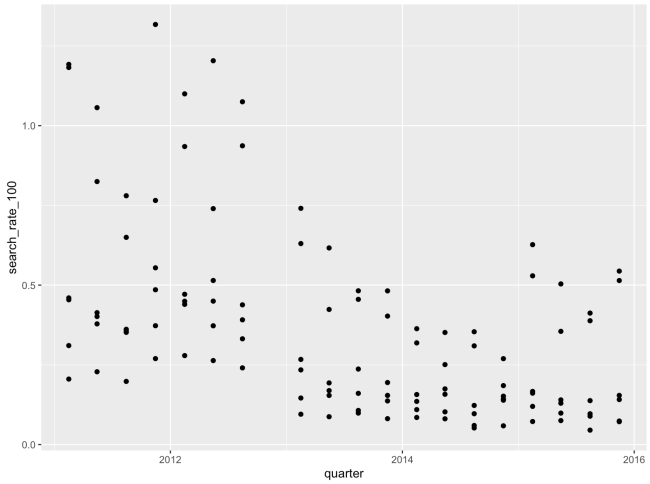
- Can specify HTML color codes

```
ggplot(data = stops,
  mapping = aes(x = quarter,
                y = search_rate_100)) +
  geom_point(color = "#63B3E8")
```



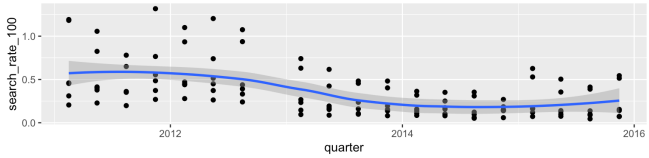
Data can be passed in

```
stops %>%
  ggplot(aes(x = quarter, y = search_rate_100)) +
    geom_point()
```



Assign ggplot() to objects for layering

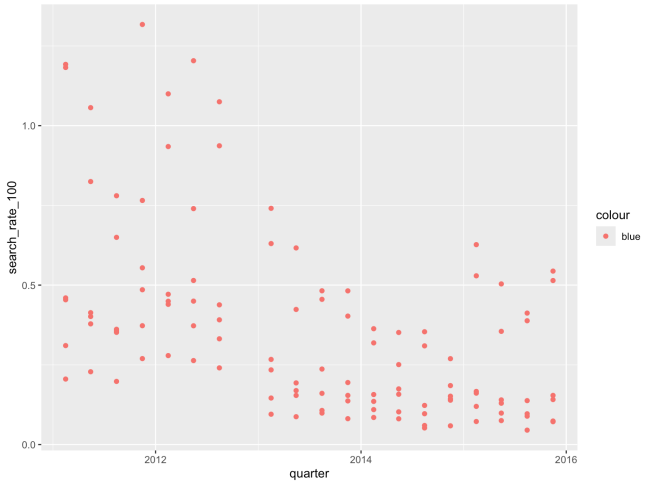
```
p <- ggplot(stops, aes(x = quarter, y = search_rate_100)) +
  geom_point()

p + geom_smooth()
```



Common early pitfalls

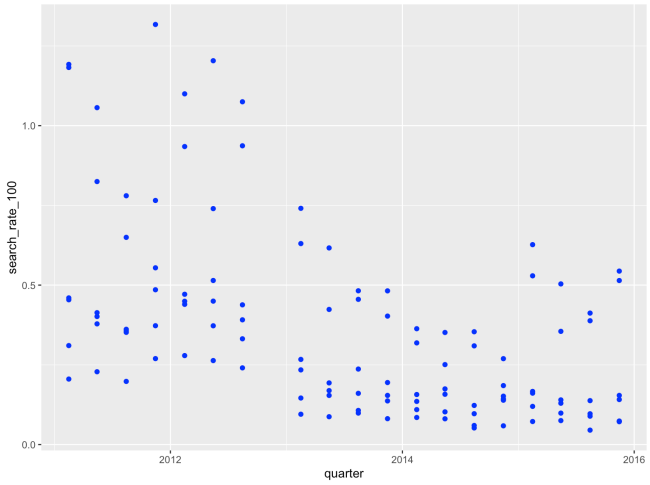
Mappings that aren't

```
ggplot(data = stops) +
  geom_point(aes(x = quarter, y = search_rate_100, color = "blue"))
```



Mappings that aren't

```
ggplot(data = stops) +
  geom_point(aes(x = quarter, y = search_rate_100), color = "blue")
```



Your turn!

Exercise: What is wrong with the following?

```
stops %>%
  ggplot(aes(x = quarter, y = search_rate_100, color = legalization_status)) %>%
    geom_point()
```

+ and %>%

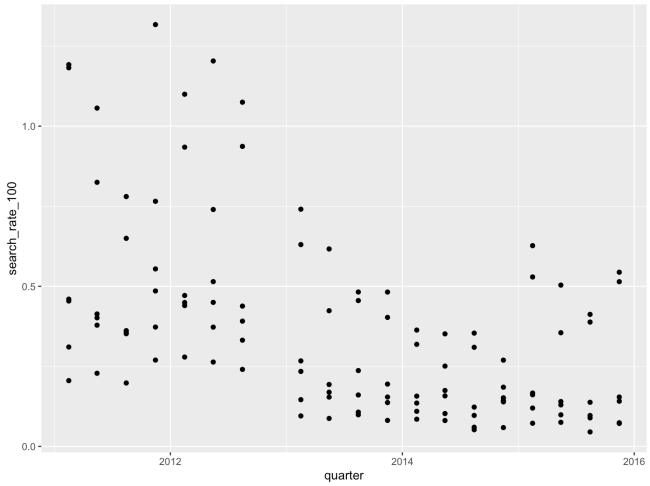
What is wrong with the following?

```
stops %>%
  ggplot(aes(x = quarter, y = search_rate_100, color = legalization_status)) %>%
    geom_point()
```

```
## Error in `geom_point()`:
## ! `mapping` must be created by `aes()`.
## ℹ Did you use `%>%` or `|>` instead of `+`?
```

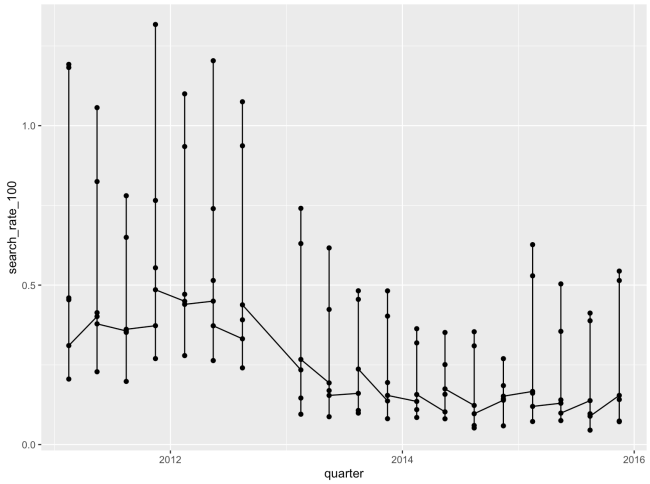
Basic plot

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100)) +
  geom_point()
```



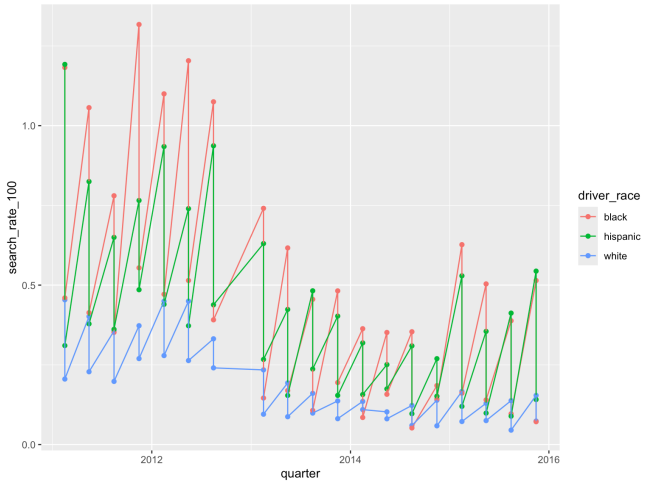
Two layers

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100)) +
  geom_point()  +
  geom_line()
```



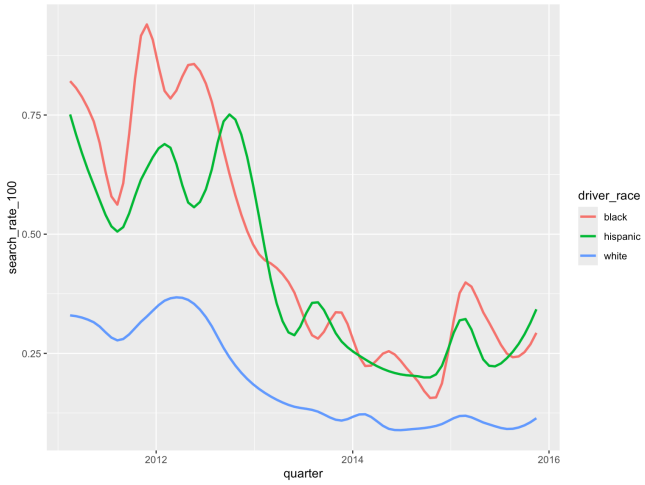
The power of groups

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  geom_line()
```



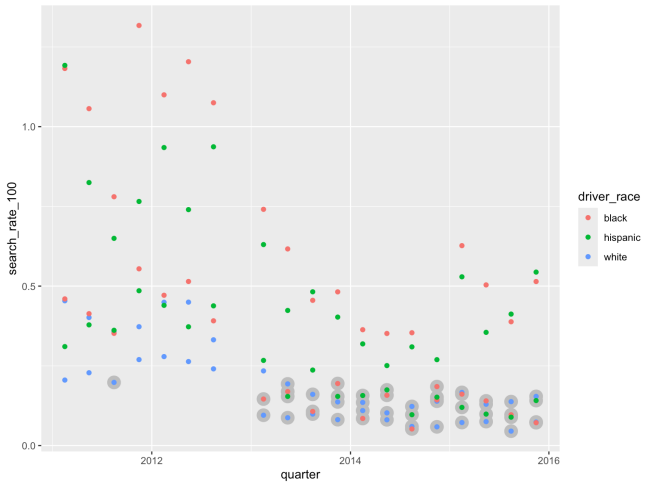
Now we've got it

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(span = 0.2, se = FALSE)
```



Control data by layer

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = filter(stops, search_rate_100 < .2),
             size = 5, color = "gray") +
  geom_point()
```



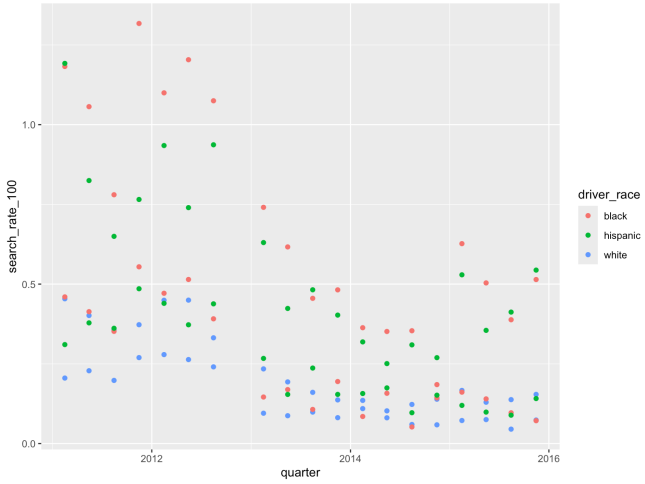
Your turn!

Exercise: Work with your neighbor to sketch what the following plots will look like. No cheating! Do not run the code, just think through the code for the time being.

```
pre_legalization_high <- stops %>%
  filter((quarter < "2013-01-01" & search_rate_100 > 1.0))
```

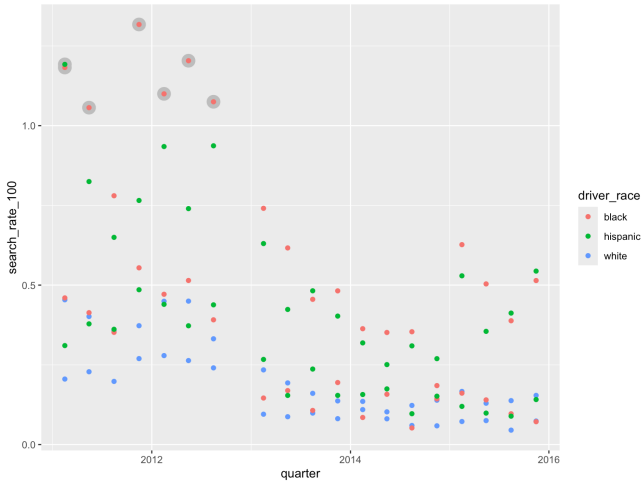
```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point() +
  geom_text(data = pre_legalization_high, aes(y = search_rate_100 + .05, label =
        search_rate_100),
            size = 2, color = "black")
```

```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point()
```

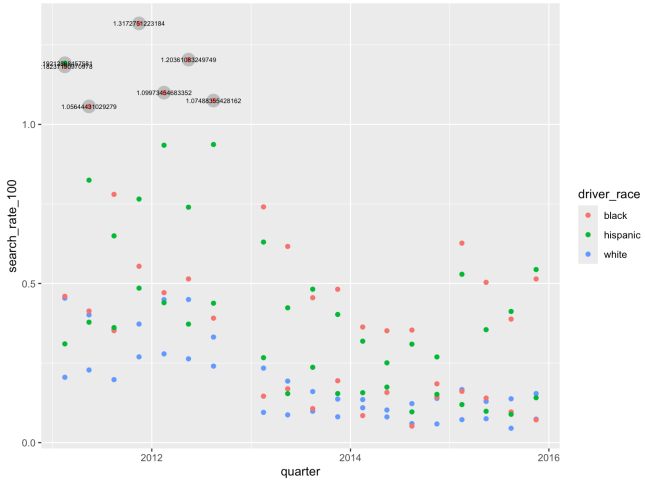


```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  geom_point(data = pre_legalization_high, size = 5, color = "gray")
```

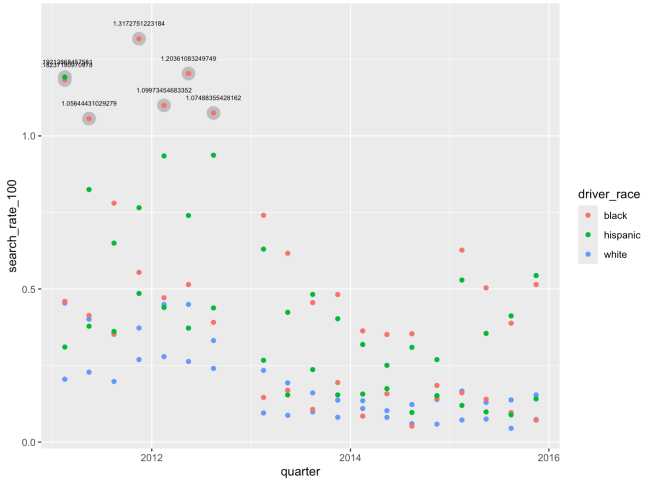
```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point()
```



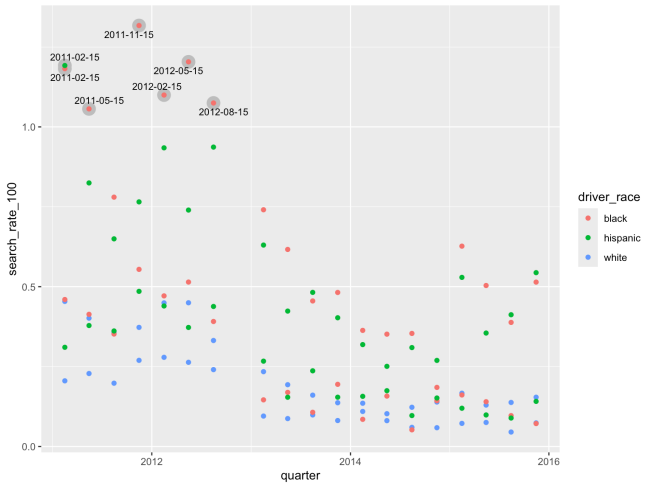
```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point() +
  geom_text(data = pre_legalization_high, aes(y = search_rate_100, label =
        search_rate_100),
            size = 2, color = "black")
```



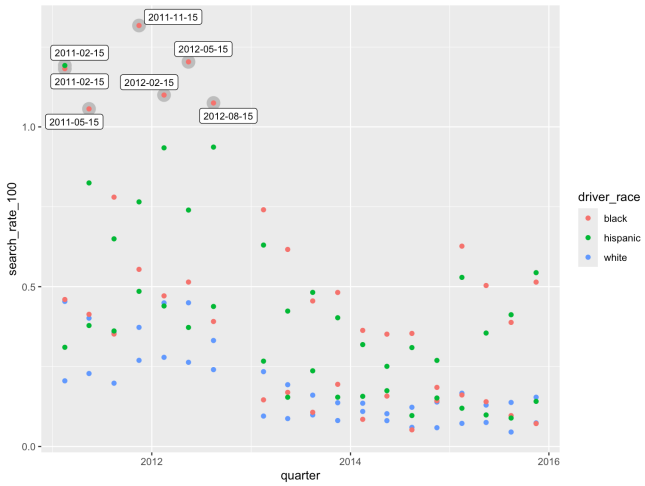
```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point() +
  geom_text(data = pre_legalization_high, aes(y = search_rate_100 + .05, label =
        search_rate_100),
            size = 2, color = "black")
```



```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point() +
  geom_text_repel(data = pre_legalization_high,
                  aes(x = quarter, y = search_rate_100,
                      label = as.character(quarter)),
                  size = 3, color = "black")
```

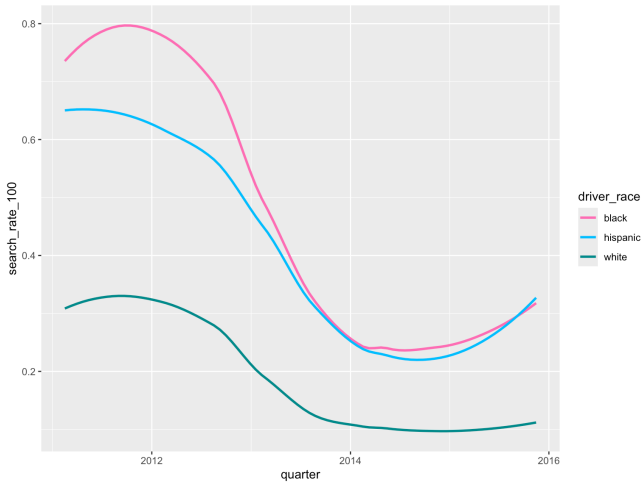


```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point(data = pre_legalization_high, size = 5, color = "gray") +
  geom_point() +
  geom_label_repel(data = pre_legalization_high,
                   aes(x = quarter, y = search_rate_100,
                       label = as.character(quarter)),
                   size = 3, color = "black")
```



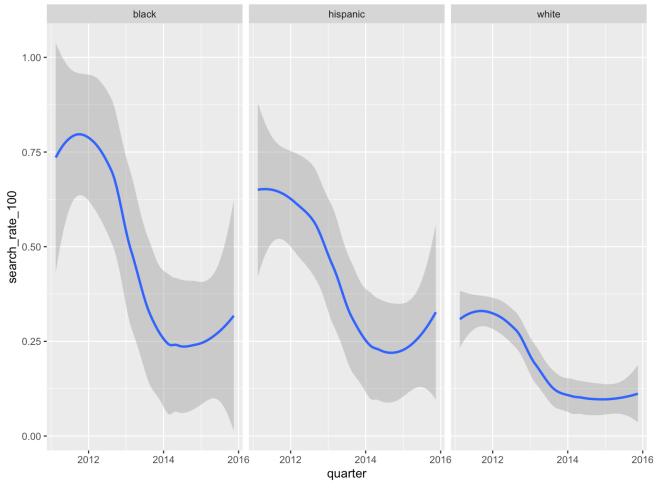
Specifying colors

```
ggplot(stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  scale_color_manual(values = c("#FF6EB4", "#00BFFF", "#008B8B")) +
  geom_smooth(se = FALSE)
```



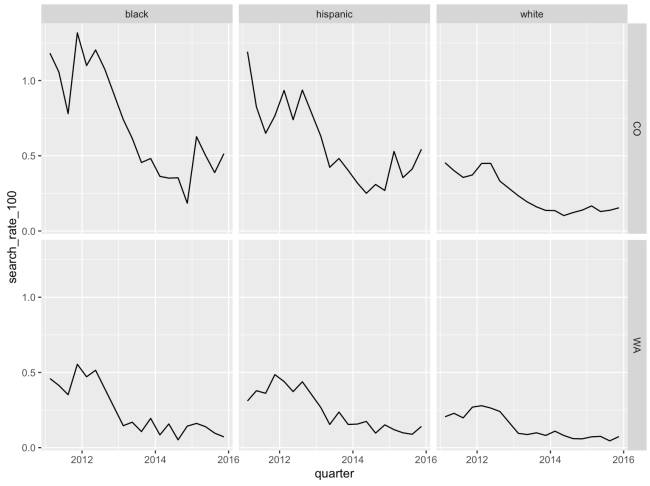
Splitting over facets

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100)) +
  geom_smooth() +
  facet_wrap( ~ driver_race)
```



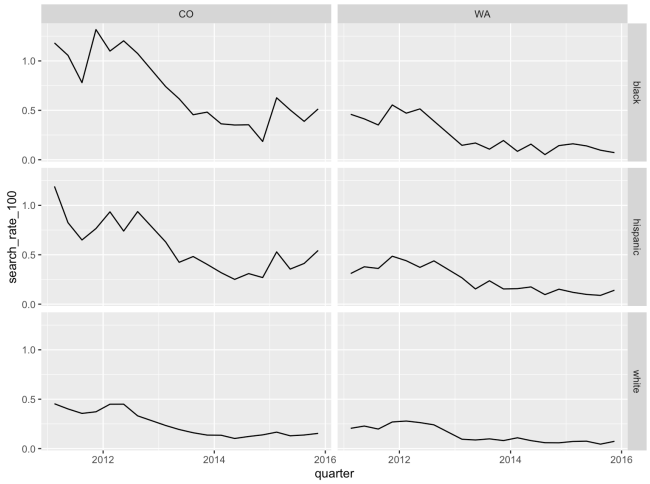
facet_grid

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100)) +
  geom_line() +
  facet_grid(state ~ driver_race)
```



facet_grid

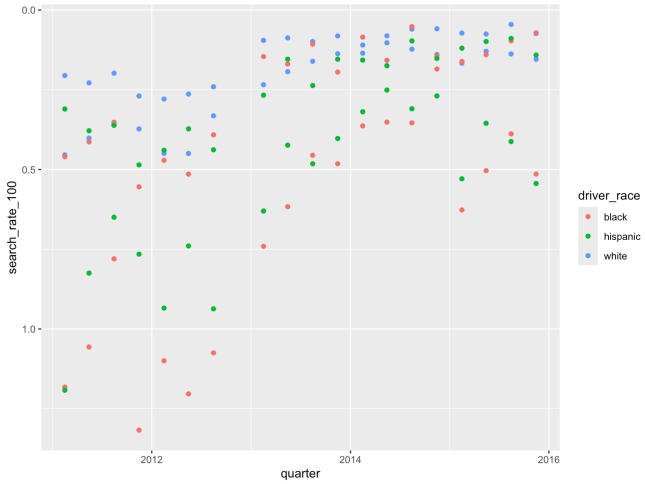
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100)) +
  geom_line() +
  facet_grid(driver_race ~ state)
```



Scales and legends

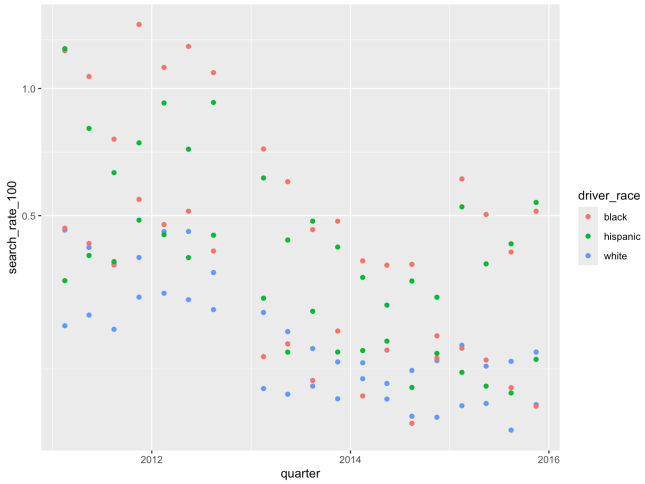
Scale transformation

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  scale_y_reverse()
```



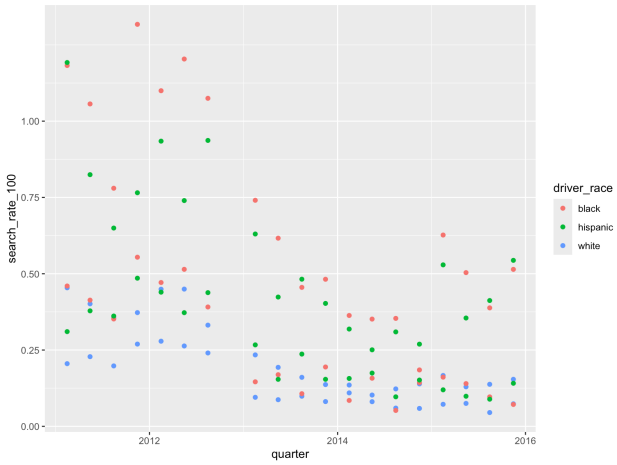
Scale transformation

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  scale_y_sqrt()
```



Scale details

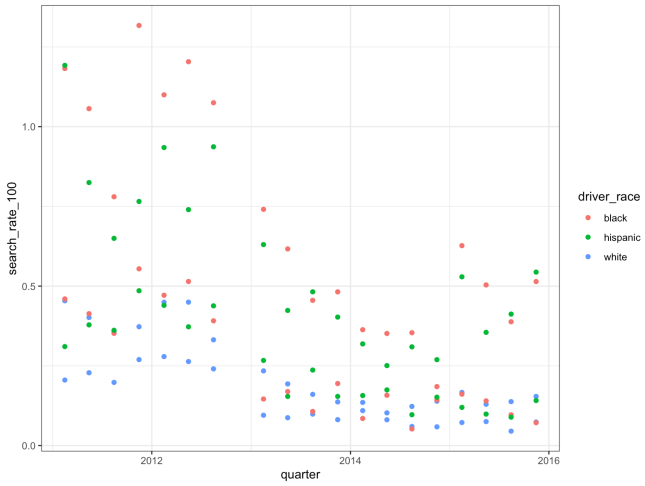
```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  scale_y_continuous(breaks = c(0, 0.25, 0.5, .75, 1.0))
```



Themes

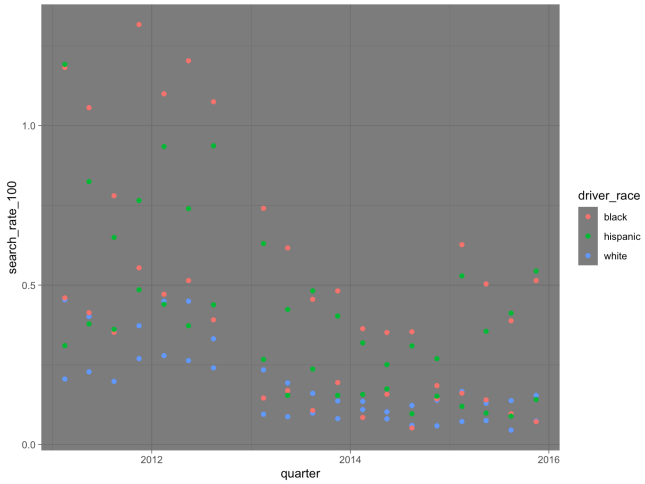
Overall themes

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  theme_bw()
```



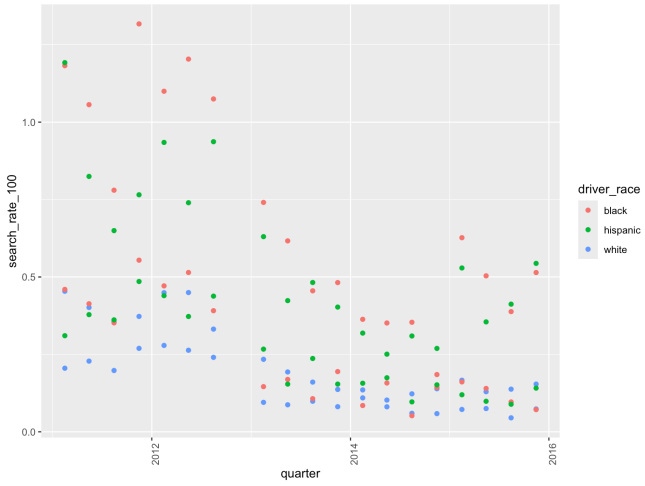
Overall themes

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  theme_dark()
```



Customizing theme elements

```
ggplot(data = stops, aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90))
```

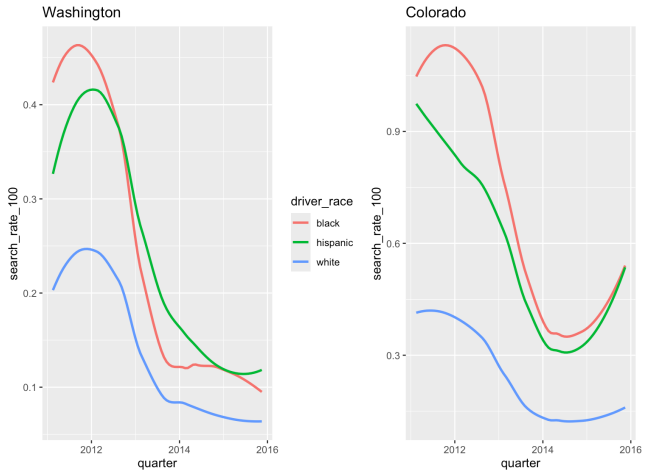


Combining several plots to a grid

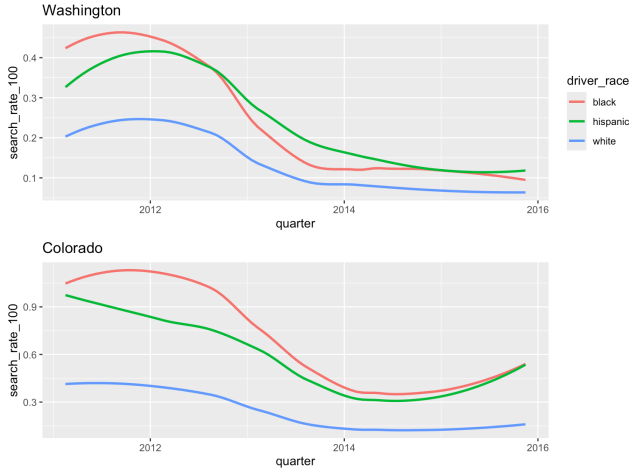
```
wa_stops <- stops %>% filter(state == "WA") %>%
  ggplot(aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(se = FALSE) +
  labs(title = "Washington")

co_stops <- stops %>% filter(state == "CO") %>%
  ggplot(aes(x = quarter, y = search_rate_100, color = driver_race)) +
  geom_smooth(se = FALSE) +
  labs(title = "Colorado") +
  theme(legend.position = "none")
```

Combining several plots to a grid

```
wa_stops + co_stops
```

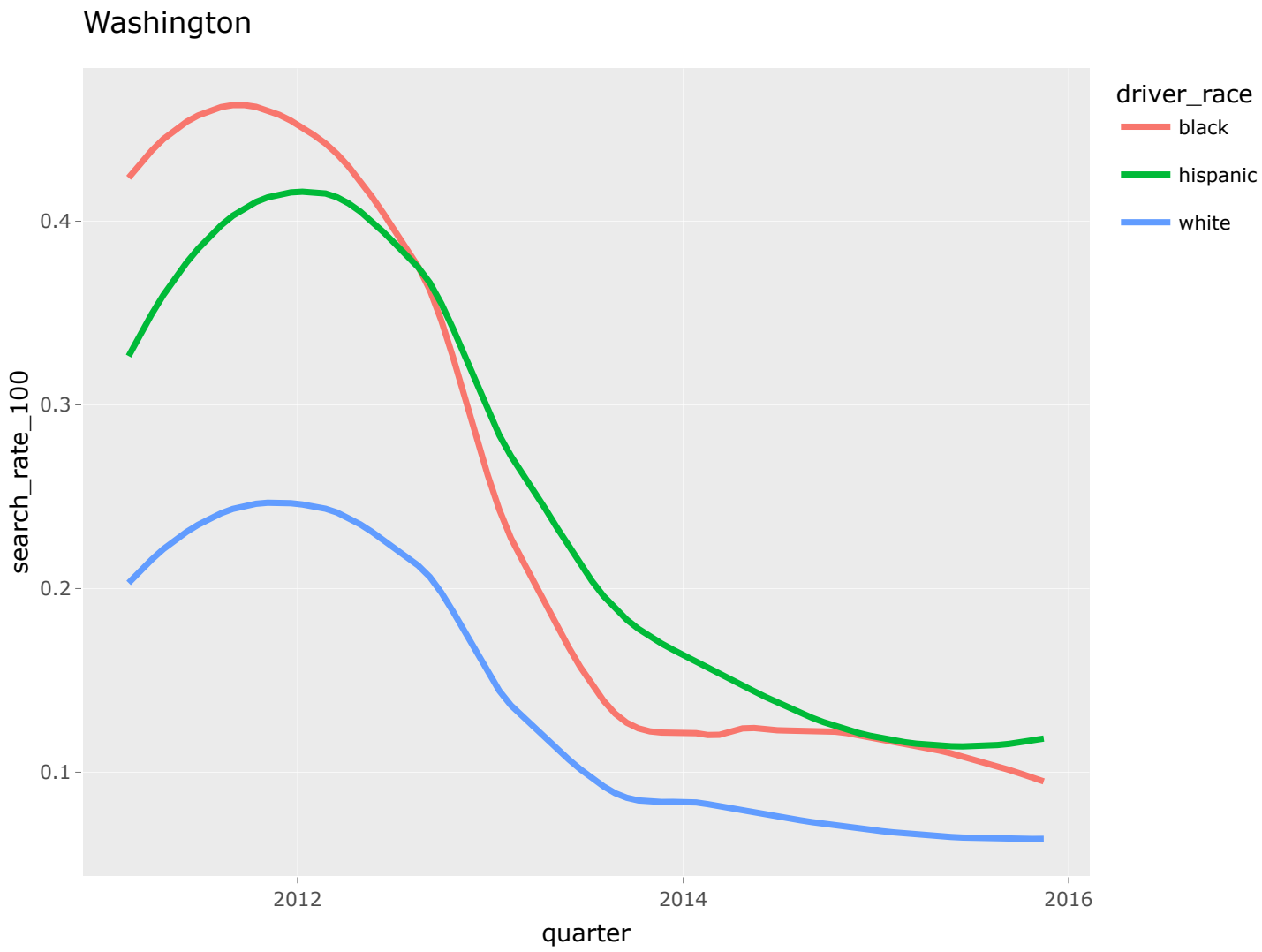



```
(wa_stops / co_stops)
```



Interactivity

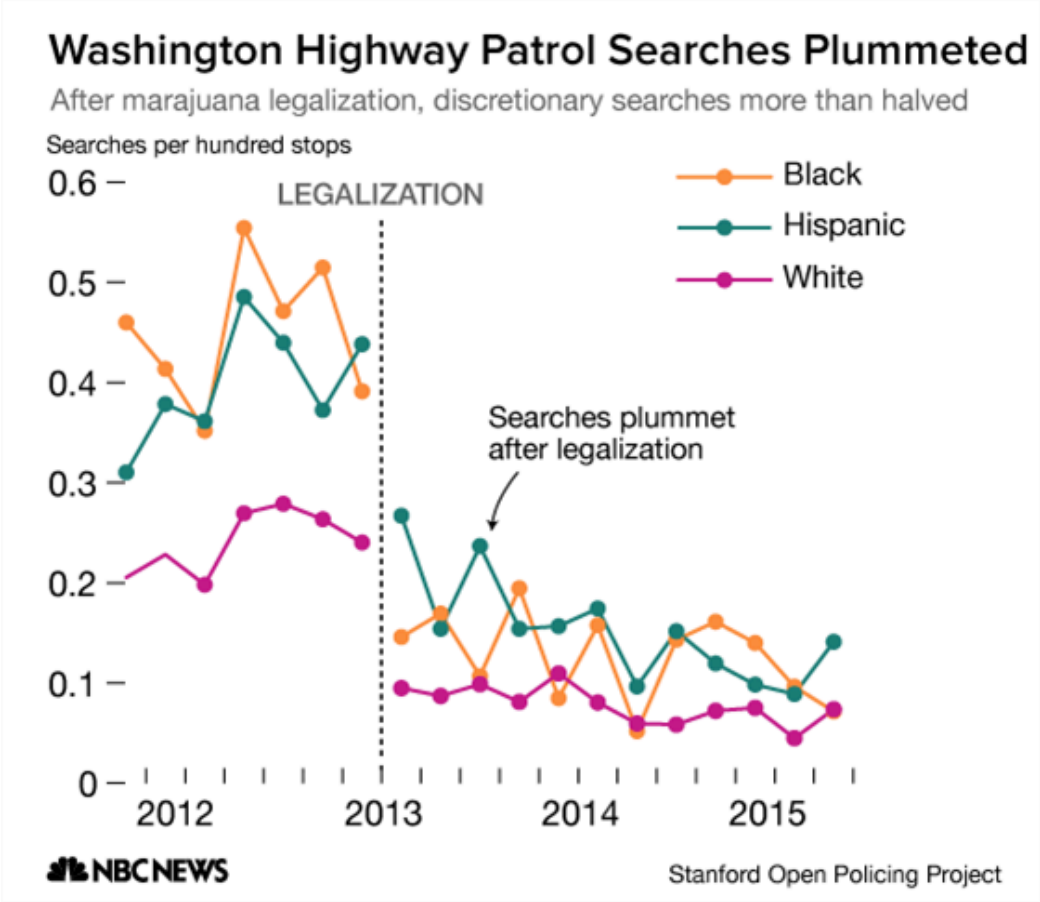
```
plotly::ggplotly(wa_stops)
```



Your turn!

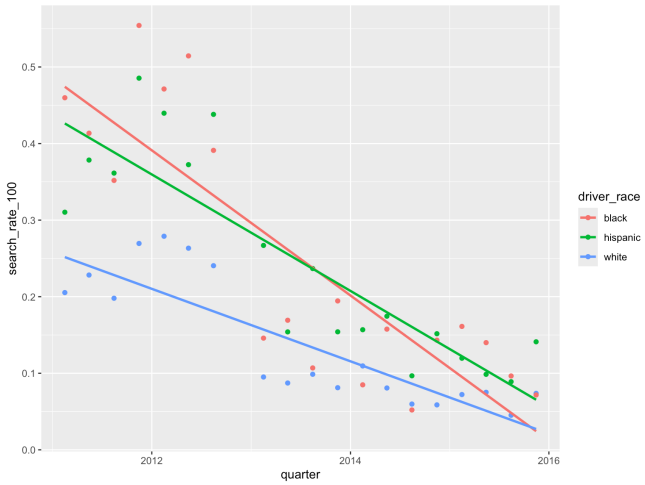
Final Exercise:

Recreate this chart



Starter code:

```
stops %>% filter(state == "WA") %>%
  ggplot(aes(quarter, search_rate_100, color = driver_race)) +
  geom_point() +
  geom_smooth(method = lm, se = FALSE)
```



- '?labs' layer controls title, subtitle, caption, etc.
- '?scale_color_manual' layer allows you to specify your own colors to the levels
- '?geom_vline' layer draws a vertical line across the plot. (hint: the x-axis is a date data type)
- '?theme' controls the non-data elements of the plot like size of text, angle of axis ticks, etc.
- '?annotate' creates a text annotation layer. Same trick with coordinates as geom_vline

- Experiment with themes

Themes Vignette

To really master themes:

ggplot2.tidyverse.org/articles/extending-ggplot2.html#creating-your-own-theme

Recap

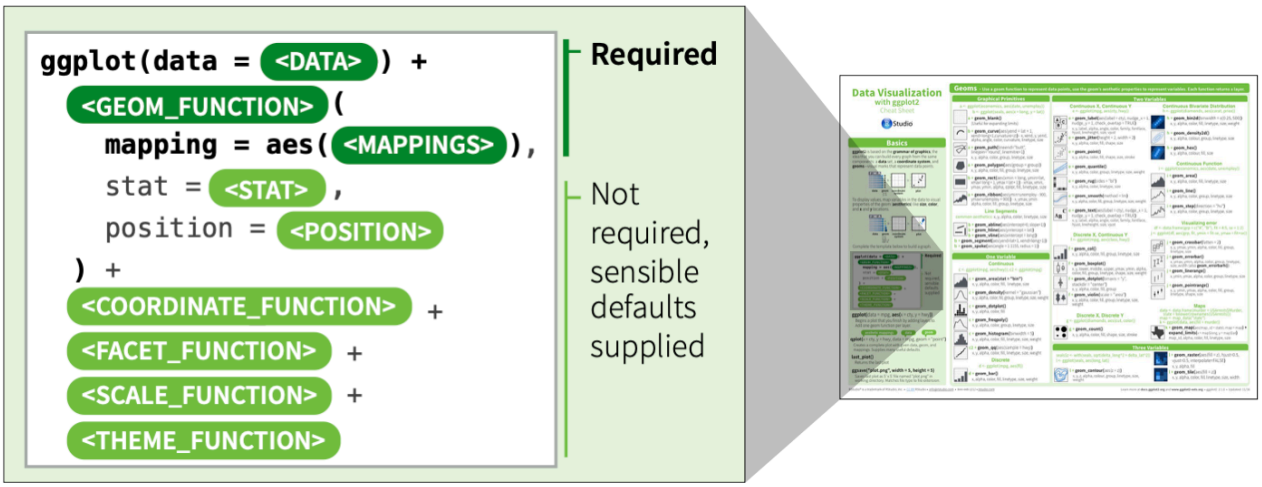
Takeaways

- map variables to aethestics
- add "geoms" for visual representation layers
- scales can be independently managed
- legends are automatically created but can be managed
- statistics are sometimes calculated by geoms so understand how geoms work

ggplot2 template

Make any plot by filling in the parameters of this template

```
knitr::include_graphics("./img/ggplot2-template.png")
```



Learn more

- Books:
  - *R for Data Science* by Grolemund and Wickham
  - *R Graphics Cookbook* by Chang
  - *Data Visualization: A Practical Introduction* by Healy
- ggplot2.tidyverse.org
- ggplot2 Cheat sheet