

# Week 1: *Getting Started*

 EMSE 4575: Exploratory Data Analysis

 John Paul Helveston

 August 31, 2022

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. Workflow & Reading In Data
5. Wrangling Data
6. Visualizing Data

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. Workflow & Reading In Data
5. Wrangling Data
6. Visualizing Data

# Course 1: Intro to Programming for Analytics

## "Computational Literacy"

- Programming: Conditionals (if/else), loops, functions, testing, data types.
- Analytics: Data structures, import / export, basic data manipulation & visualization.

# Course 2: Exploratory Data Analysis

## "Data Literacy"

- Strategies for conducting an exploratory data analysis.
- Design principles for visualizing and communicating *information* extracted from data.
- Reproducibility: Reports that contain code, equations, visualizations, and narrative text.

**Class goal:** translate *data* into *information*

# Class goal: translate *data* into *information*

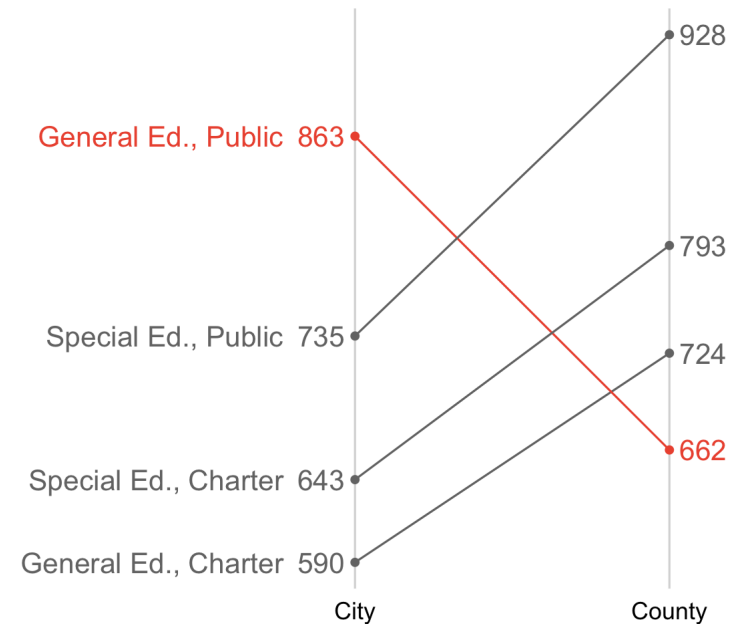
## Data

Average student engagement scores

Class	Type	City	County
Special Ed.	Charter	643	793
Special Ed.	Public	735	928
General Ed.	Charter	590	724
General Ed.	Public	863	662

## Information

Students in public, general education classes in county schools have surprisingly low engagement



# Data exploration: an iterative process

Encode data:

```
engagement_data <- data.frame(  
  City = c(643, 735, 590, 863),  
  County = c(793, 928, 724, 662),  
  School = c('Special Ed., Charter', 'Special Ed., Pu  
             'General Ed., Charter', 'General Ed., Pu  
engagement_data
```

```
#>   City County      School  
#> 1  643   793 Special Ed., Charter  
#> 2  735   928 Special Ed., Public  
#> 3  590   724 General Ed., Charter  
#> 4  863   662 General Ed., Public
```

Re-format data for plotting:

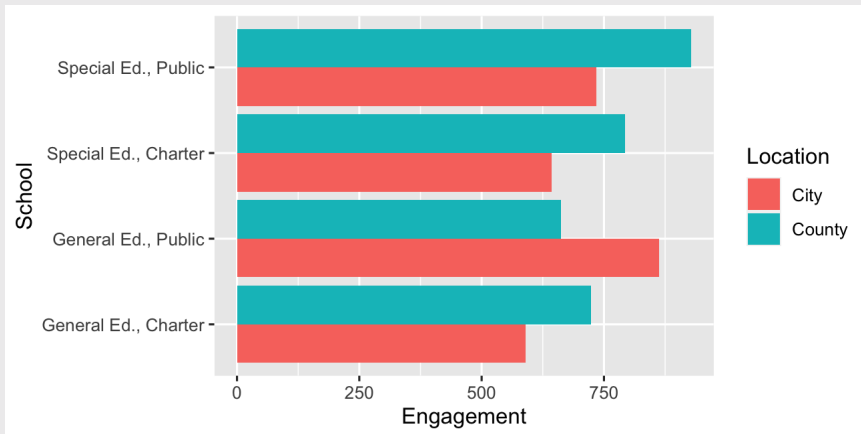
```
engagement_data <- engagement_data %>%  
  gather(Location, Engagement, City:County) %>%  
  mutate(Location = fct_relevel(  
    Location, c('City', 'County'))  
engagement_data
```

```
#>      School Location Engagement  
#> 1 Special Ed., Charter      City      643  
#> 2 Special Ed., Public      City      735  
#> 3 General Ed., Charter      City      590  
#> 4 General Ed., Public      City      863  
#> 5 Special Ed., Charter     County      793  
#> 6 Special Ed., Public     County      928  
#> 7 General Ed., Charter     County      724  
#> 8 General Ed., Public     County      662
```

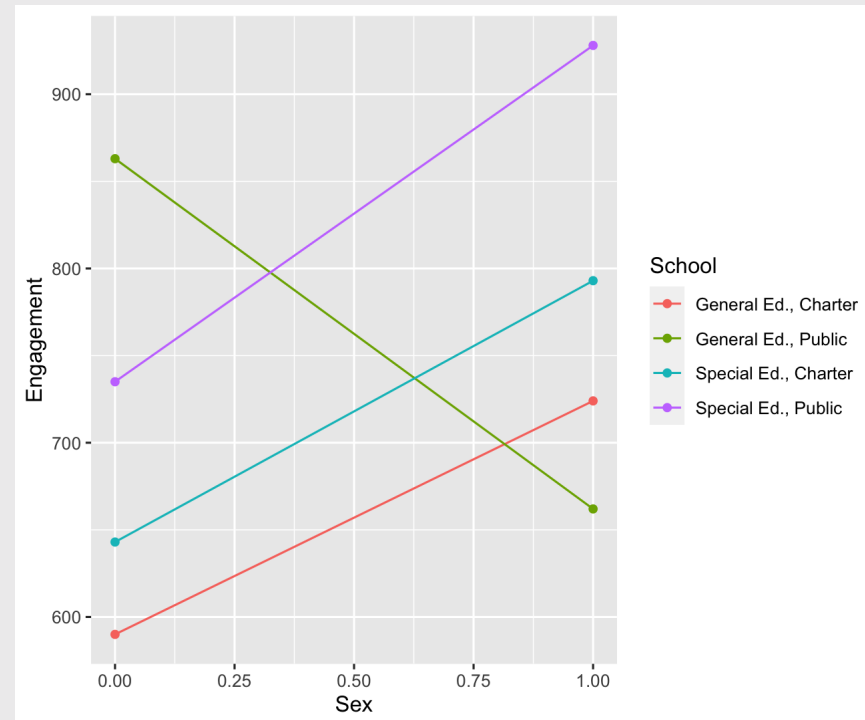
# Data exploration: an iterative process

Initial exploratory plotting:

```
engagement_data %>%  
  ggplot() +  
  geom_col(aes(x = Engagement, y = School,  
              fill = Location),  
          position = 'dodge')
```



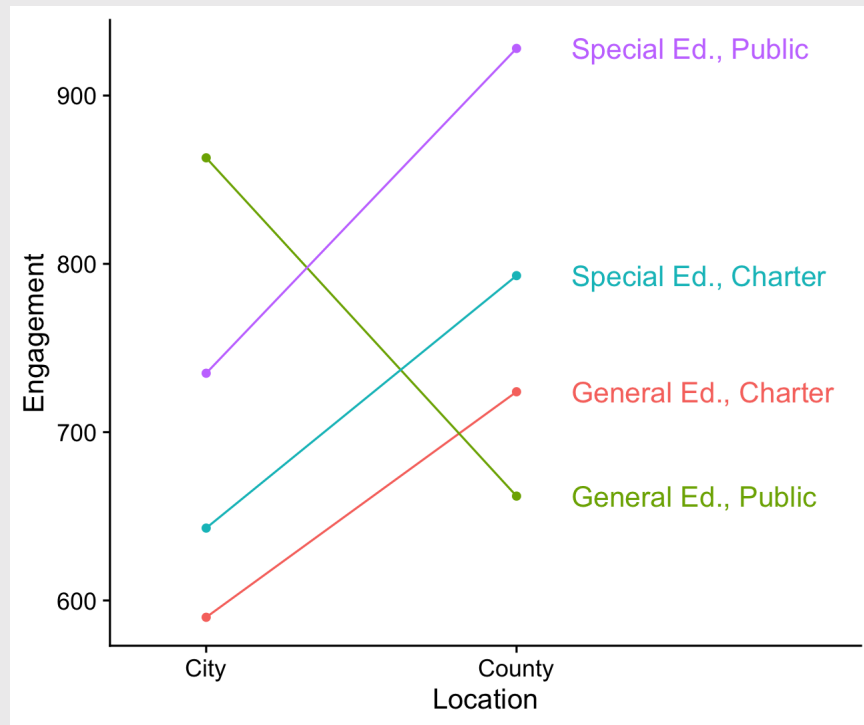
More exploratory plotting:  
highlight difference



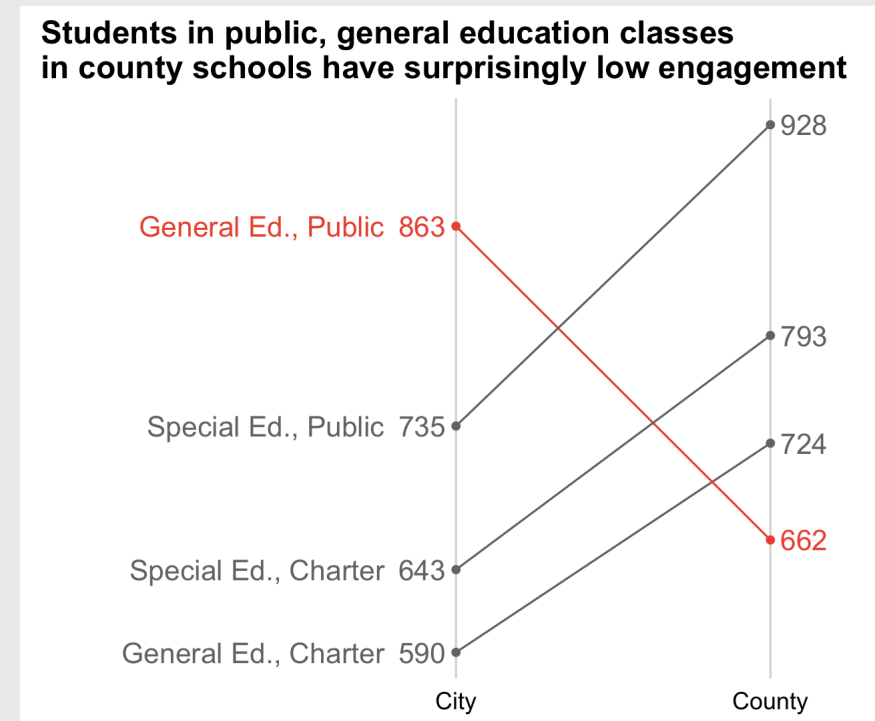


# Data exploration: an iterative process

Directly label figure:



Remove unnecessary axes, change colors, fix labels:



# A fully reproducible analysis

Code

Plot

```
data <- data.frame(
  City = c(643, 735, 590, 863),
  County = c(793, 928, 724, 662),
  School = c('Special Ed., Charter', 'Special Ed., Public',
             'General Ed., Charter', 'General Ed., Public'),
  Highlight = c(0, 0, 0, 1)) %>%
gather(Location, Engagement, City:County) %>%
mutate(
  Location = fct_relevel(Location, c('City', 'County')),
  Highlight = as.factor(Highlight),
  x = ifelse(Location == 'County', 1, 0))
```

```
plot <- ggplot(data, aes(x = x, y = Engagement, group = School, color = Highlight))
  geom_point() +
  geom_line() +
  scale_color_manual(values = c('#757575', '#ed573e')) +
  labs(x = 'Sex', y = 'Engagement',
       title = paste0('Students in public, general education classes\n',
                      'in county schools have surprisingly low engagement')) +
  scale_x_continuous(limits = c(-1.2, 1.2), labels = c('City', 'County'),
                    breaks = c(0, 1)) +
  geom_text_repel(aes(label = Engagement, color = as.factor(Highlight)),
                 data = subset(engagement, Location == 'County'),
                 size = 5,
                 nudge_x = 0.1,
                 segment.color = NA) +
  geom_text_repel(aes(label = Engagement, color = as.factor(Highlight)),
                 data = subset(engagement, Location == 'City'),
                 size = 5,
                 nudge_x = -0.1,
                 segment.color = NA) +
  geom_text_repel(aes(label = School, color = as.factor(Highlight)),
                 data = subset(engagement, Location == 'City'),
                 size = 5,
                 nudge_x = -0.25,
                 hjust = 1,
                 segment.color = NA) +
  theme_cowplot() +
  background_grid(major = 'x') +
  theme(axis.line = element_blank(),
        axis.title.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks = element_blank(),
        legend.position = 'none')
```

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. Workflow & Reading In Data
5. Wrangling Data
6. Visualizing Data

# Meet your instructor!



John Helveston, Ph.D.

- 2018 - Present Assistant Professor, Engineering Management & Systems Engineering
- 2016-2018 Postdoc at [Institute for Sustainable Energy](#), Boston University
- 2016 PhD in Engineering & Public Policy at Carnegie Mellon University
- 2015 MS in Engineering & Public Policy at Carnegie Mellon University
- 2010 BS in Engineering Science & Mechanics at Virginia Tech
- Website: [www.jhelvy.com](http://www.jhelvy.com)

# Meet your tutors!



## **Michael Rossetti**

- Graduate Assistant (GA)
- PhD student in EMSE

# Meet your tutors!



## **Eliese Ottinger**

- Learning Assistant (LA)
- EMSE Senior & P4A / EDA alumni

# Prerequisites


## EMSE 4574: Intro to Programming for Analytics

You should be able to:

- Use RStudio to write basic R commands.
- Know the distinctions between different R operators and data types, including numeric, string, and logical data.
- Use **tidyverse** functions to wrangle and manipulate data in R.
- Use the **ggplot2** library to create plots in R.

 Check out R for Analytics Primer

# Course website

 Everything you need will be on the course website:  
<https://eda.seas.gwu.edu/2022-Fall/>

 The [schedule](#) is the best starting point



# Quizzes




📅 At the start of class every other week-ish, unscheduled. Make ups only for excused absences (i.e. don't be late).

📅 5 total, lowest dropped

🕒 ~5 - 10 minutes

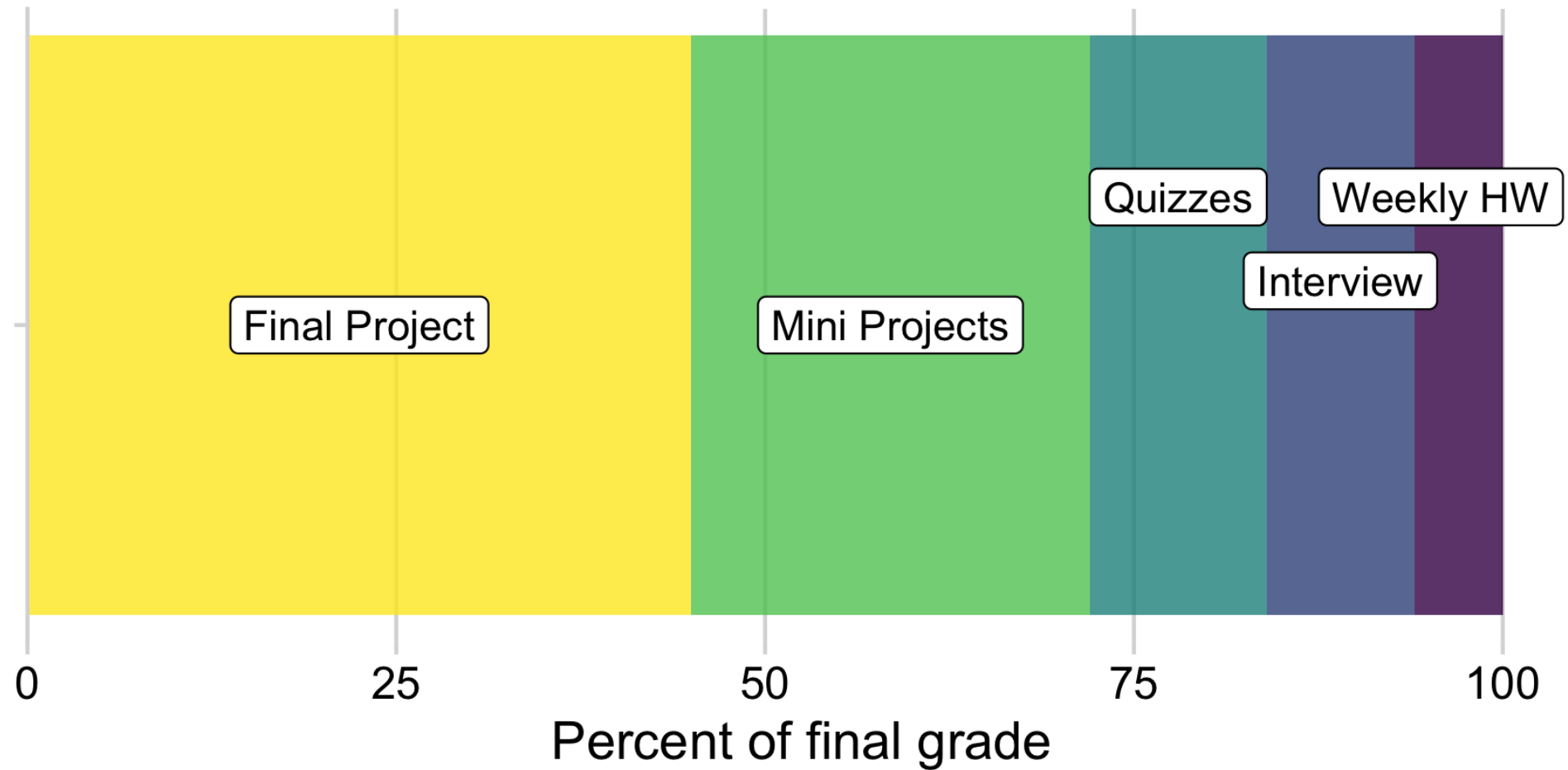
**Why quiz at all?** The "retrieval effect" - basically, you have to *practice* remembering things, otherwise your brain won't remember them (see the book "[Make It Stick: The Science of Successful Learning](#)")

# Assignments

- 1)  Weekly Homework / Readings: **HW1**
- 2)  3 Mini Projects (due 2 weeks from date assigned)
- 3)  **Final Project** (Teams of 2 - 3 students)

Item	Due Date
Proposal	March 12
Progress Report	April 16
Final Report	April 30
Presentation	May 03
Interview	Exam week

# Grades



# Grades

Item	Weight	Notes
Weekly HW	12 %	
Quizzes	8 %	5 quizzes, lowest dropped
Mini Project 1	8 %	Individual assignments
Mini Project 2	8 %	
Mini Project 3	8 %	
Final Project: Proposal	9 %	Teams of 2-3 students
Final Project: Progress Report	12 %	
Final Project: Report	16 %	
Final Project: Presentation	9 %	
Final Interview	10 %	Individual interview

# Course policies

- BE NICE
- BE HONEST
- DON'T CHEAT

Copying is good, stealing is bad

"Plagiarism is trying to pass someone else's work off as your own. Copying is about reverse-engineering."

-- Austin Kleon, from [Steal Like An Artist](#)

# Late submissions

- **5** late days - use them anytime, no questions asked
- No more than **2** late days on any one assignment
- Contact me for special cases

# How to succeed in this class

 Participate during class!


 Start assignments early and **read carefully!**

 Actually read (before class)!

 Get sleep and take breaks often!

 Ask for help!

# Getting Help

 Use **Slack** to ask questions.

 Meet with your tutors

 **Schedule a meeting** w/ Prof. Helveston:

- Mondays from 8:00-5:00pm
- Wednesdays from 3:20-5:00pm
- Thursdays from 12:00-5:00pm


 **GW Coders**



## Course Software

 **Slack**: See bb for link to join;  
install on phone and **turn notifications on!**

 **R** & **RStudio** (Install both)

 **RStudio Cloud** (Register for free!)

*Break*

Install Stuff

05 : 00

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. **Workflow & Reading In Data**
5. Wrangling Data
6. Visualizing Data

# Workflow for reading in data

1) Use R Projects (.Rproj files) to organize your analysis - **don't double-click .R files!**



2) Use the [here](#) package to create file paths

```
path <- here::here("folder", "file.csv")
```

3) Import data with these functions:

File type	Function	Library
.csv	read_csv()	readr
.txt	read_table()	utils
.xlsx	read_excel()	readxl

# Importing Comma Separated Values (.csv)

Read in `.csv` files with `read_csv()`:

```
library(tidyverse)
library(here)

csvPath <- here('data', 'milk_production.csv')
milk_production <- read_csv(csvPath)

head(milk_production)
```

```
#> # A tibble: 6 × 4
#>   region    state    year milk_produced
#>   <chr>    <chr>    <dbl>         <dbl>
#> 1 Northeast Maine      1970      619000000
#> 2 Northeast New Hampshire 1970      356000000
#> 3 Northeast Vermont    1970     1970000000
#> 4 Northeast Massachusetts 1970      658000000
#> 5 Northeast Rhode Island 1970       75000000
#> 6 Northeast Connecticut 1970      661000000
```

# Importing Text Files (.txt)

Read in `.txt` files with `read.table()`:

```
txtPath <- here('data', 'nasa_global_temps.txt')
global_temps <- read.table(txtPath, skip = 5, header = FALSE)
head(global_temps)
```

```
#>      V1      V2      V3
#> 1 1880 -0.15 -0.08
#> 2 1881 -0.07 -0.12
#> 3 1882 -0.10 -0.15
#> 4 1883 -0.16 -0.19
#> 5 1884 -0.27 -0.23
#> 6 1885 -0.32 -0.25
```

# Importing Text Files (.txt)

Read in `.txt` files with `read.table()`:

```
txtPath <- here('data', 'nasa_global_temps.txt')
global_temps <- read.table(txtPath, skip = 5, header = FALSE)
names(global_temps) <- c('year', 'no_smoothing', 'loess') # Add header
head(global_temps)
```

```
#>   year no_smoothing loess
#> 1 1880      -0.15 -0.08
#> 2 1881      -0.07 -0.12
#> 3 1882      -0.10 -0.15
#> 4 1883      -0.16 -0.19
#> 5 1884      -0.27 -0.23
#> 6 1885      -0.32 -0.25
```

# Importing Excel Files (.xlsx)

Read in `.xlsx` files with `read_excel()`:

```
library(readxl)
```

```
xlsxPath <- here('data', 'pv_cell_production.xlsx')
```

```
pv_cells <- read_excel(xlsxPath, sheet = 'Cell Prod by Country', skip = 2)
```

```
glimpse(pv_cells)
```

```
#> Rows: 25
#> Columns: 10
#> $ Year      <chr> NA, NA, "1995", "1996", "1997", "1998", "1999", "2000", "2001", "2002"
#> $ China     <chr> "Megawatts", NA, "NA", "NA", "NA", "NA", "NA", "NA", "2.5", "3", "10", "13"
#> $ Taiwan    <chr> NA, NA, "NA", "NA", "NA", "NA", "NA", "NA", "3.5", "8", "17", "39.299"
#> $ Japan     <dbl> NA, NA, 16.4, 21.2, 35.0, 49.0, 80.0, 128.6, 171.2, 251.1, 363.9, 601
#> $ Malaysia  <chr> NA, NA, "NA", "NA", "NA", "NA", "NA", "NA", "0", "0", "0", "0", "0",
#> $ Germany    <chr> NA, NA, "NA", "NA", "NA", "NA", "NA", "NA", "22.5", "23.5", "55", "121.5",
#> $ `South Korea` <chr> NA, NA, "NA", "NA", "NA", "NA", "NA", "NA", "0", "0", "0", "0", "5.3"
#> $ `United States` <dbl> NA, NA, 34.7500, 38.8500, 51.0000, 53.7000, 60.8000, 75.0000, 100.300
#> $ Others     <chr> NA, NA, "NA", "NA", "NA", "NA", "NA", "NA", "48.200000000000017", "69.80000
#> $ World      <dbl> NA, NA, 77.600, 88.600, 125.800, 154.900, 201.300, 276.800, 371.300,
```



# Importing Excel Files (.xlsx)

Read in `.xlsx` files with `read_excel()`:

```
library(readxl)

xlsxPath <- here('data', 'pv_cell_production.xlsx')
pv_cells <- read_excel(xlsxPath, sheet = 'Cell Prod by Country', skip = 2) %>%
  mutate(Year = as.numeric(Year)) %>% # Convert "non-years" to NA
  filter(!is.na(Year)) # Drop NA rows in Year
```

```
glimpse(pv_cells)
```

```
#> Rows: 19
#> Columns: 10
#> $ Year      <dbl> 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 20
#> $ China     <chr> "NA", "NA", "NA", "NA", "NA", "2.5", "3", "10", "13", "40", "128.30000000000001", "34
#> $ Taiwan    <chr> "NA", "NA", "NA", "NA", "NA", "NA", "3.5", "8", "17", "39.299999999999997", "88", "16
#> $ Japan     <dbl> 16.4, 21.2, 35.0, 49.0, 80.0, 128.6, 171.2, 251.1, 363.9, 601.5, 833.0, 926.4, 937.5,
#> $ Malaysia  <chr> "NA", "NA", "NA", "NA", "NA", "NA", "0", "0", "0", "0", "0", "0", "100.1", "397.9", "
#> $ Germany   <chr> "NA", "NA", "NA", "NA", "NA", "22.5", "23.5", "55", "121.5", "193", "339", "469.1", "
#> $ `South Korea` <chr> "NA", "NA", "NA", "NA", "NA", "NA", "0", "0", "0", "0", "5.3", "13", "31.883935905674
#> $ `United States` <dbl> 34.7500, 38.8500, 51.0000, 53.7000, 60.8000, 75.0000, 100.3000, 120.6000, 103.0000, 1
#> $ Others    <chr> "NA", "NA", "NA", "NA", "NA", "48.200000000000017", "69.800000000000011", "97.2999999
#> $ World     <dbl> 77.600, 88.600, 125.800, 154.900, 201.300, 276.800, 371.300, 542.000, 749.400, 1198.80
```

# Your turn

10:00

Open the `practice.Rmd` file.

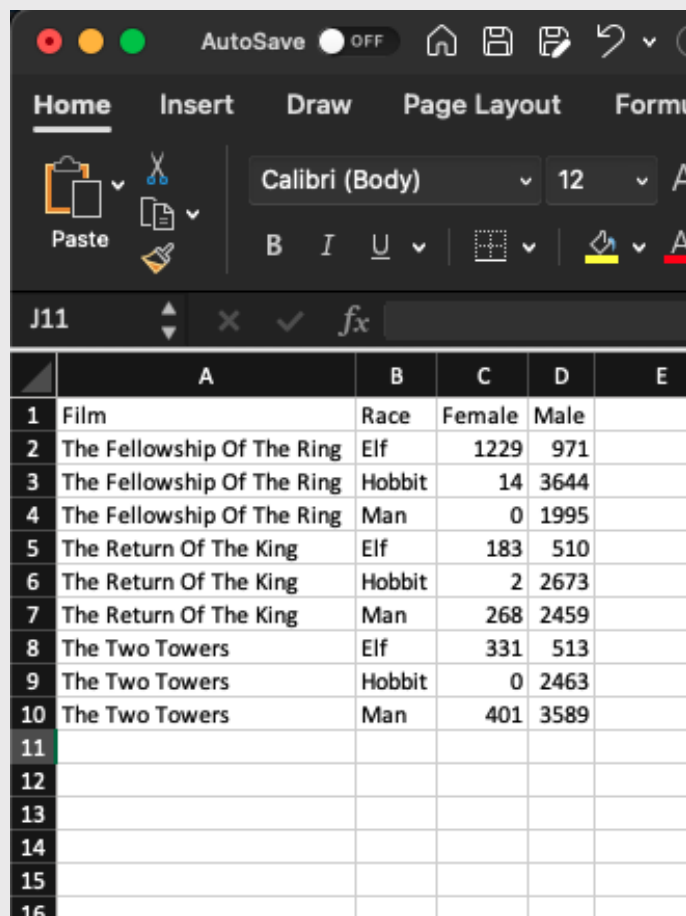
Write code to import the following data files from the "data" folder:

- For `lotr_words.csv`, call the data frame `lotr`
- For `north_america_bear_killings.txt`, call the data frame `bears`
- For `uspto_clean_energy_patents.xlsx`, call the data frame `patents`

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. Workflow & Reading In Data
5. **Wrangling Data**
6. Visualizing Data

# The data frame... in Excel



A screenshot of the Microsoft Excel application. The ribbon at the top shows 'Home', 'Insert', 'Draw', 'Page Layout', and 'Formulas'. The 'Home' ribbon is active, showing options for 'Paste', font settings (Calibri (Body), size 12), and text formatting (Bold, Italic, Underline). The formula bar shows 'J11'. The spreadsheet grid shows columns A through E and rows 1 through 16. The data is as follows:

	A	B	C	D	E
1	Film	Race	Female	Male	
2	The Fellowship Of The Ring	Elf	1229	971	
3	The Fellowship Of The Ring	Hobbit	14	3644	
4	The Fellowship Of The Ring	Man	0	1995	
5	The Return Of The King	Elf	183	510	
6	The Return Of The King	Hobbit	2	2673	
7	The Return Of The King	Man	268	2459	
8	The Two Towers	Elf	331	513	
9	The Two Towers	Hobbit	0	2463	
10	The Two Towers	Man	401	3589	
11					
12					
13					
14					
15					
16					

# The data frame... in R

lotr

```
#> # A tibble: 18 × 4
#>   film      race  gender
#>   <chr>    <chr>  <chr>
#> 1 The Fellowship Of The Ring Elf      Female
#> 2 The Fellowship Of The Ring Elf      Male
#> 3 The Fellowship Of The Ring Hobbit   Female
#> 4 The Fellowship Of The Ring Hobbit   Male
#> 5 The Fellowship Of The Ring Man      Female
#> 6 The Fellowship Of The Ring Man      Male
#> 7 The Return Of The King    Elf      Female
#> 8 The Return Of The King    Elf      Male
#> 9 The Return Of The King    Hobbit   Female
#> 10 The Return Of The King    Hobbit   Male
#> 11 The Return Of The King    Man      Female
#> 12 The Return Of The King    Man      Male
#> 13 The Two Towers           Elf      Female
```

# Columns: *Vectors* of values (must be same data type)

Extract a column using `$`

```
lotr$race
```

```
#> [1] "Elf"      "Elf"      "Hobbit"   "Hobbit"   "Man"      "Man"      "Elf"      "Elf"      "Hobbit"   "
```

# Columns: *Vectors* of values (must be same data type)

Can also use brackets:

```
lotr$race
```

```
#> [1] "Elf" "Elf" "Hobbit" "Hobbit" "Man" "Man" "Elf" "Elf" "Hobbit" "
```

```
lotr[,2]
```

```
#> # A tibble: 18 × 1  
#>   race  
#>   <chr>  
#> 1 Elf  
#> 2 Elf  
#> 3 Hobbit  
#> 4 Hobbit  
#> 5 Man  
#> 6 Man  
#> 7 Elf  
#> 8 Elf
```

# Rows: Information about individual observations

Information about the first row:

```
lotr[1,]
```

```
#> # A tibble: 1 × 4  
#>   film          race gender word_count  
#>   <chr>        <chr> <chr>     <dbl>  
#> 1 The Fellowship Of The Ring Elf      Female      1229
```

Information about rows 1 & 2:

```
lotr[1:2,]
```

```
#> # A tibble: 2 × 4  
#>   film          race gender word_count  
#>   <chr>        <chr> <chr>     <dbl>  
#> 1 The Fellowship Of The Ring Elf      Female      1229  
#> 2 The Fellowship Of The Ring Elf      Male        971
```

# Quick Practice

Read in the `data.csv` file in the "data" folder:

```
data <- read_csv(here('data', 'data.csv'))
```

Now answer these questions:

- How many rows and columns are in the data frame?
- What type of data is each column?
- Preview the different columns - what do you think this data is about? What might one row represent?
- How many unique airlines are in the data frame?
- What is the shortest and longest air time for any one flight in the data frame?



The tidyverse: `stringr` + `dplyr` + `readr` + `ggplot2` + ...



Art by [Allison Horst](#)

# The main `dplyr` "verbs"

"Verb"	What it does
<code>select()</code>	Select columns by name
<code>filter()</code>	Keep rows that match criteria
<code>arrange()</code>	Sort rows based on column(s)
<code>mutate()</code>	Create new columns
<code>summarize()</code>	Create summary values

# Core `tidyverse` concept: **Chain functions together with "pipes"**

`%>%`

Think of the words "...and then..."

```
data %>%  
  do_something() %>%  
  do_something_else()
```

Select columns with `select()`

## Subset Variables (Columns)



# Select columns with `select()`

Select the columns `film` & `race`

```
lotr %>%  
  select(film, race)
```

```
#> # A tibble: 18 × 2  
#>   film                race  
#>   <chr>             <chr>  
#> 1 The Fellowship Of The Ring Elf  
#> 2 The Fellowship Of The Ring Elf  
#> 3 The Fellowship Of The Ring Hobbit  
#> 4 The Fellowship Of The Ring Hobbit  
#> 5 The Fellowship Of The Ring Man  
#> 6 The Fellowship Of The Ring Man  
#> 7 The Return Of The King      Elf  
#> 8 The Return Of The King      Elf  
#> 9 The Return Of The King      Hobbit  
#> 10 The Return Of The King      Hobbit  
#> 11 The Return Of The King      Man  
#> 12 The Return Of The King      Man
```

# Select columns with `select()`

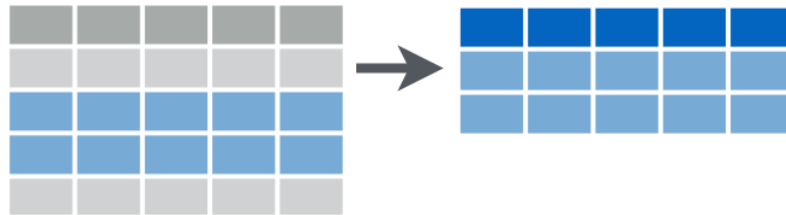
Use the `-` sign to drop columns

```
lotr %>%  
  select(-film)
```

```
#> # A tibble: 18 × 3  
#>   race    gender word_count  
#>   <chr>  <chr>      <dbl>  
#> 1 Elf    Female      1229  
#> 2 Elf    Male        971  
#> 3 Hobbit Female        14  
#> 4 Hobbit Male      3644  
#> 5 Man    Female         0  
#> 6 Man    Male      1995  
#> 7 Elf    Female       183  
#> 8 Elf    Male       510  
#> 9 Hobbit Female         2  
#> 10 Hobbit Male     2673  
#> 11 Man    Female       268  
#> 12 Man    Male     2459
```

# Filter for rows with `filter()`

## Subset Observations (Rows)



# Filter for rows with `filter()`

Keep only the rows with Elf characters

```
lotr %>%  
  filter(race == "Elf")
```

```
#> # A tibble: 6 × 4  
#>   film          race gender word_count  
#>   <chr>      <chr> <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Elf    Female    1229  
#> 2 The Fellowship Of The Ring Elf    Male      971  
#> 3 The Return Of The King    Elf    Female    183  
#> 4 The Return Of The King    Elf    Male     510  
#> 5 The Two Towers            Elf    Female    331  
#> 6 The Two Towers            Elf    Male     513
```



# Filter for rows with `filter()`

Keep only the rows with Elf or Hobbit characters

```
lotr %>%  
  filter((race == "Elf") | (race == "Hobbit"))
```

```
#> # A tibble: 12 × 4  
#>   film      race gender word_count  
#>   <chr>    <chr>  <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Elf      Female    1229  
#> 2 The Fellowship Of The Ring Elf      Male      971  
#> 3 The Fellowship Of The Ring Hobbit   Female     14  
#> 4 The Fellowship Of The Ring Hobbit   Male    3644  
#> 5 The Return Of The King    Elf      Female    183  
#> 6 The Return Of The King    Elf      Male     510  
#> 7 The Return Of The King    Hobbit   Female      2  
#> 8 The Return Of The King    Hobbit   Male    2673  
#> 9 The Two Towers           Elf      Female    331  
#> 10 The Two Towers           Elf      Male     513  
#> 11 The Two Towers           Hobbit   Female      0  
#> 12 The Two Towers           Hobbit   Male    2463
```

# Filter for rows with `filter()`

Keep only the rows with Elf or Hobbit characters

```
lotr %>%  
  filter(race %in% c("Elf", "Hobbit"))
```

```
#> # A tibble: 12 × 4  
#>   film          race gender word_count  
#>   <chr>         <chr> <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Elf      Female      1229  
#> 2 The Fellowship Of The Ring Elf      Male        971  
#> 3 The Fellowship Of The Ring Hobbit    Female        14  
#> 4 The Fellowship Of The Ring Hobbit    Male     3644  
#> 5 The Return Of The King    Elf      Female      183  
#> 6 The Return Of The King    Elf      Male       510  
#> 7 The Return Of The King    Hobbit    Female         2  
#> 8 The Return Of The King    Hobbit    Male     2673  
#> 9 The Two Towers           Elf      Female      331  
#> 10 The Two Towers           Elf      Male       513  
#> 11 The Two Towers           Hobbit    Female         0  
#> 12 The Two Towers           Hobbit    Male     2463
```

# Logic operators for `filter()`

Description	Example
Values greater than 1	<code>value &gt; 1</code>
Values greater than or equal to 1	<code>value &gt;= 1</code>
Values less than 1	<code>value &lt; 1</code>
Values less than or equal to 1	<code>value &lt;= 1</code>
Values equal to 1	<code>value == 1</code>
Values not equal to 1	<code>value != 1</code>
Values in the set <code>c(1, 4)</code>	<code>value %in% c(1, 4)</code>

# Combine `filter()` and `select()`

Keep only the rows with Elf characters that spoke more than 1000 words, then select everything but the race column

```
lotr %>%  
  filter((race == "Elf") & (word_count > 1000)) %>%  
  select(-race)
```

```
#> # A tibble: 1 × 3  
#>   film                gender word_count  
#>   <chr>              <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Female      1229
```

Create new variables with `mutate()`

## Make New Variables



# Create new variables with `mutate()`

Create a new variable, `word1000` which is `TRUE` if the character spoke 1,000 or more words

```
lotr %>%  
  mutate(word1000 = word_count >= 1000)
```

```
#> # A tibble: 18 × 5  
#>   film      race gender word_count word1000  
#>   <chr>    <chr> <chr>      <dbl> <lgl>  
#> 1 The Fellowship Of The Ring Elf      Female      1229 TRUE  
#> 2 The Fellowship Of The Ring Elf      Male        971 FALSE  
#> 3 The Fellowship Of The Ring Hobbit    Female        14 FALSE  
#> 4 The Fellowship Of The Ring Hobbit    Male      3644 TRUE  
#> 5 The Fellowship Of The Ring Man       Female         0 FALSE  
#> 6 The Fellowship Of The Ring Man       Male      1995 TRUE  
#> 7 The Return Of The King    Elf      Female      183 FALSE  
#> 8 The Return Of The King    Elf      Male       510 FALSE  
#> 9 The Return Of The King    Hobbit   Female         2 FALSE  
#> 10 The Return Of The King   Hobbit   Male      2673 TRUE  
#> 11 The Return Of The King   Man      Female      268 FALSE
```

# Handling if/else conditions

`ifelse(<condition>, <if TRUE>, <else>)`

```
lotr %>%  
  mutate(word1000 = ifelse(word_count >= 1000, TRUE, FALSE))
```

```
#> # A tibble: 18 × 5  
#>   film          race gender word_count word1000  
#>   <chr>         <chr> <chr>      <dbl> <lgl>  
#> 1 The Fellowship Of The Ring Elf      Female    1229 TRUE  
#> 2 The Fellowship Of The Ring Elf      Male      971 FALSE  
#> 3 The Fellowship Of The Ring Hobbit   Female     14 FALSE  
#> 4 The Fellowship Of The Ring Hobbit   Male    3644 TRUE  
#> 5 The Fellowship Of The Ring Man      Female     0 FALSE  
#> 6 The Fellowship Of The Ring Man      Male    1995 TRUE  
#> 7 The Return Of The King    Elf      Female    183 FALSE  
#> 8 The Return Of The King    Elf      Male     510 FALSE  
#> 9 The Return Of The King    Hobbit   Female     2 FALSE  
#> 10 The Return Of The King    Hobbit   Male    2673 TRUE  
#> 11 The Return Of The King    Man      Female    268 FALSE
```

# Sort data frame with `arrange()`

Sort the `lotr` data frame by `word_count`

```
lotr %>%  
  arrange(word_count)
```

```
#> # A tibble: 18 × 4  
#>   film          race gender word_count  
#>   <chr>         <chr> <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Man    Female      0  
#> 2 The Two Towers Hobbit Female      0  
#> 3 The Return Of The King Hobbit Female      2  
#> 4 The Fellowship Of The Ring Hobbit Female     14  
#> 5 The Return Of The King Elf     Female    183  
#> 6 The Return Of The King Man     Female    268  
#> 7 The Two Towers Elf     Female    331  
#> 8 The Two Towers Man     Female    401  
#> 9 The Return Of The King Elf     Male     510  
#> 10 The Two Towers Elf     Male     513  
#> 11 The Fellowship Of The Ring Elf     Male     971  
#> 12 The Fellowship Of The Ring Elf     Female   1229
```



# Sort data frame with `arrange()`

Use the `desc()` function to sort in descending order

```
lotr %>%  
  arrange(desc(word_count))
```

```
#> # A tibble: 18 × 4  
#>   film          race gender word_count  
#>   <chr>         <chr> <chr>      <dbl>  
#> 1 The Fellowship Of The Ring Hobbit Male      3644  
#> 2 The Two Towers      Man   Male      3589  
#> 3 The Return Of The King  Hobbit Male      2673  
#> 4 The Two Towers      Hobbit Male      2463  
#> 5 The Return Of The King  Man   Male      2459  
#> 6 The Fellowship Of The Ring Man   Male      1995  
#> 7 The Fellowship Of The Ring Elf   Female    1229  
#> 8 The Fellowship Of The Ring Elf   Male      971  
#> 9 The Two Towers      Elf   Male      513  
#> 10 The Return Of The King Elf   Male      510  
#> 11 The Two Towers      Man   Female     401  
#> 12 The Two Towers      Elf   Female     331
```

# Your turn

10:00

Read in the `data.csv` file in the "data" folder:

```
data <- read_csv(here('data', 'data.csv'))
```

Now answer these questions:

- Create a new data frame, `flights_fall`, that contains only flights that departed in the fall semester.
- Create a new data frame, `flights_dc`, that contains only flights that flew to DC airports (Reagan or Dulles).
- Create a new data frame, `flights_dc_carrier`, that contains only flights that flew to DC airports (Reagan or Dulles) and only the columns about the month and airline.
- How many unique airlines were flying to DC airports in July?
- Create a new variable, `speed`, in miles per hour using the `time` (minutes) and `distance` (miles) variables.
- Which flight flew the fastest?
- Remove rows that have `NA` for `air_time` and re-arrange the resulting data frame based on the longest air time and longest flight distance.

# Week 1: *Getting Started*

1. Course Goal
2. Course Introduction
3. Break: Install Stuff
4. Workflow & Reading In Data
5. Wrangling Data
6. Visualizing Data

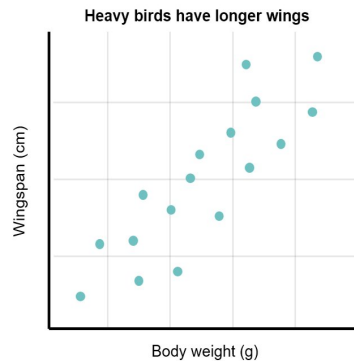
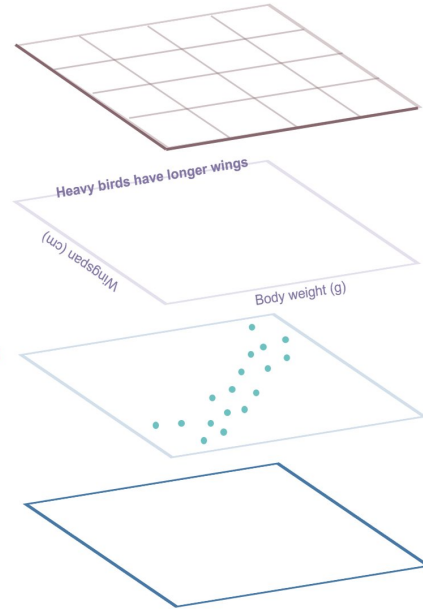
## MAKING A GRAPH WITH GGPLOT2

Customise the look of your plot with themes  
(pre-made or your own!):  
`+ theme_bw()`

Add labels and titles:  
`+ labs(x = "Body weight (g)", y = "Wingspan (cm)",  
title = "Heavy birds have longer wings")`

Specify the type of graph and the variables to use:  
`+ geom_point(aes(x = body.weight, y = wingspan))`

Plot the device containing your data:  
`ggplot(data = birds)`



# "Grammar of Graphics"

Concept developed by Leland Wilkinson  
(1999)

**ggplot2** package developed by Hadley  
Wickham (2005)

# Making plot layers with ggplot2

1. The data
2. The aesthetic mapping (what goes on the axes?)
3. The geometries (points? bars? etc.)
4. The annotations / labels
5. The theme

# Layer 1: The data

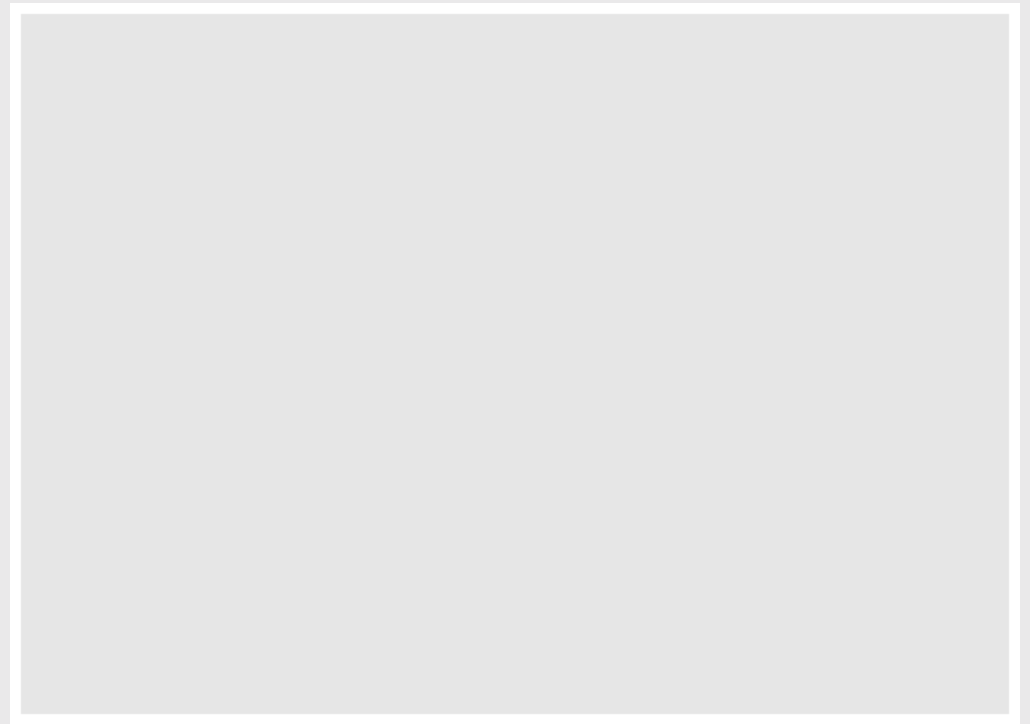
```
head(mpg)
```

```
#> # A tibble: 6 × 11
#>   manufacturer model displ  year   cyl trans      drv   cty   hwy fl   class
#>   <chr>         <chr> <dbl> <int> <int> <chr>    <chr> <int> <int> <chr> <chr>
#> 1 audi         a4      1.8  1999     4 auto(l5)  f      18    29 p    compact
#> 2 audi         a4      1.8  1999     4 manual(m5) f      21    29 p    compact
#> 3 audi         a4      2    2008     4 manual(m6) f      20    31 p    compact
#> 4 audi         a4      2    2008     4 auto(av)   f      21    30 p    compact
#> 5 audi         a4      2.8  1999     6 auto(l5)  f      16    26 p    compact
#> 6 audi         a4      2.8  1999     6 manual(m5) f      18    26 p    compact
```

# Layer 1: The data

The `ggplot()` function initializes the plot with whatever data you're using

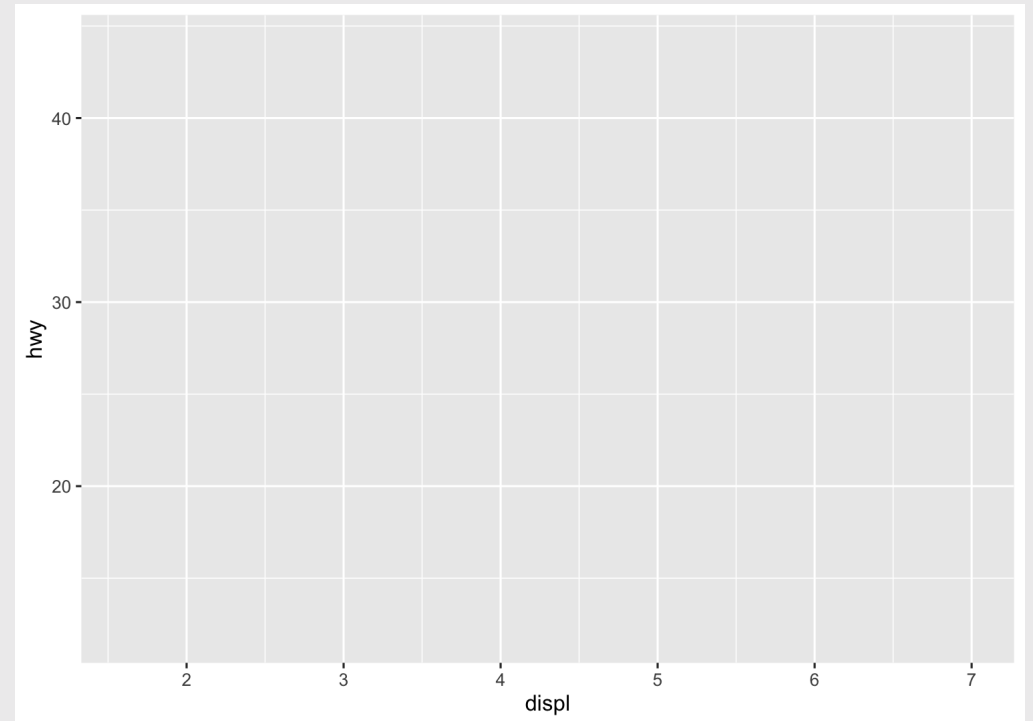
```
mpg %>%  
  ggplot()
```



# Layer 2: The aesthetic mapping

The `aes()` function determines which variables will be *mapped* to the geometries (e.g. the axes)

```
mpg %>%  
  ggplot(aes(x = displ, y = hwy))
```

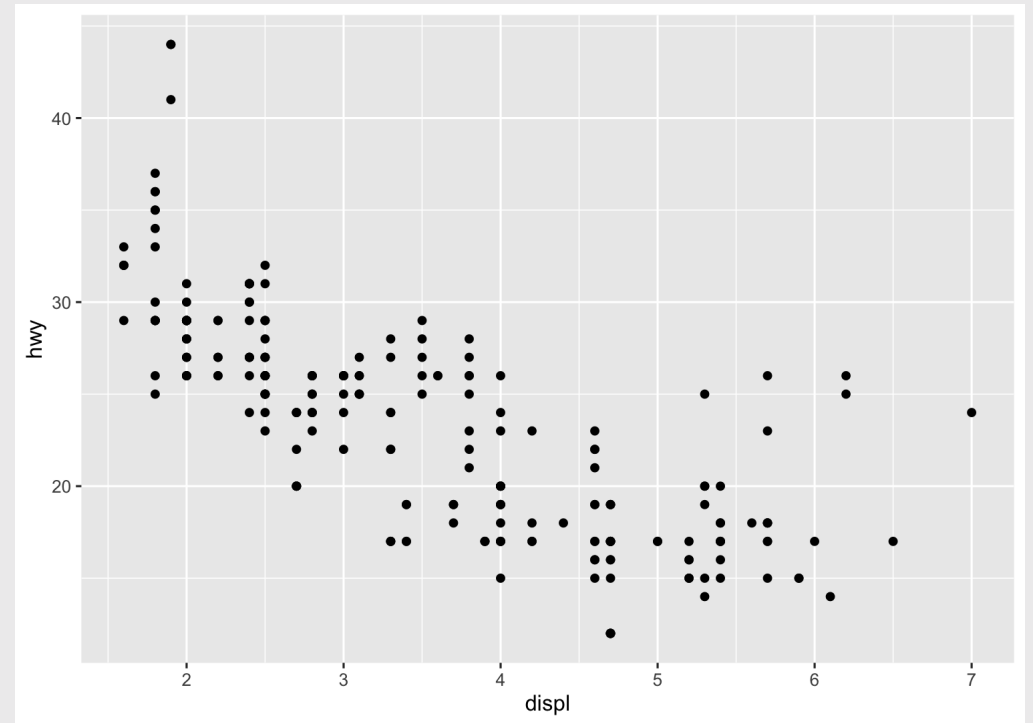




# Layer 3: The geometries

Use `+` to add geometries, e.g. `geom_points()` for points

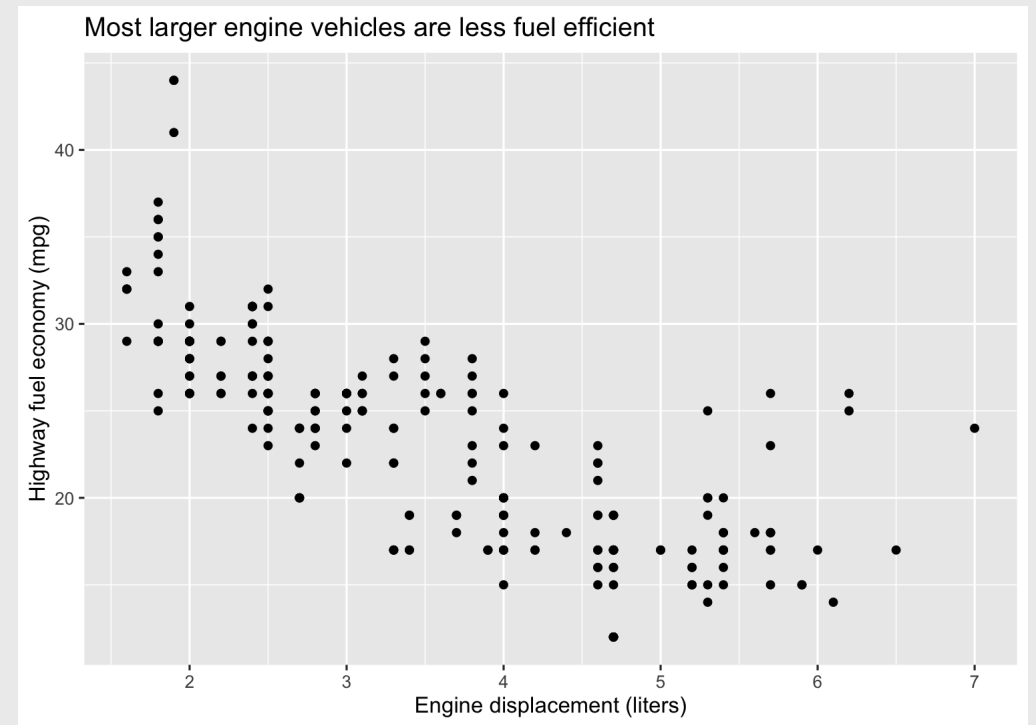
```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point()
```



# Layer 4: The annotations / labels

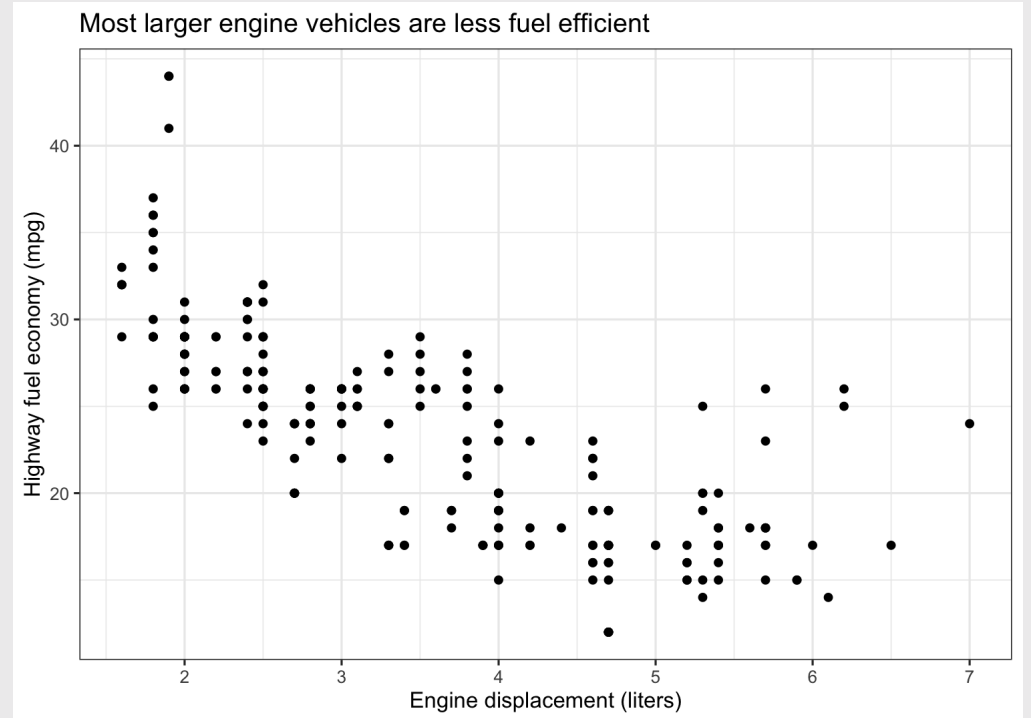
Use `labs()` to modify most labels

```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  labs(  
    x = "Engine displacement (liters)",  
    y = "Highway fuel economy (mpg)",  
    title = "Most larger engine vehicles are less fuel efficient"  
  )
```



# Layer 5: The theme

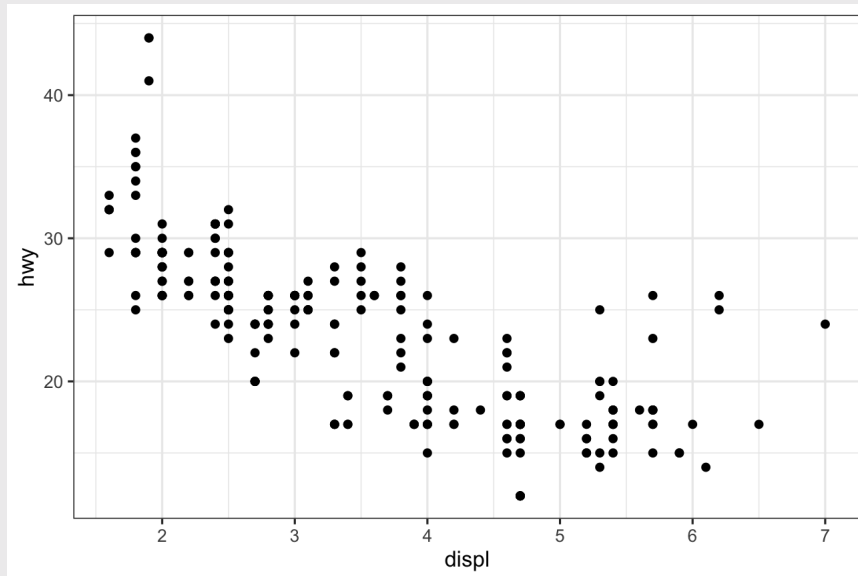
```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  labs(  
    x = "Engine displacement (liters)",  
    y = "Highway fuel economy (mpg)",  
    title = "Most larger engine vehicles are  
  ) +  
  theme_bw()
```



# Common themes

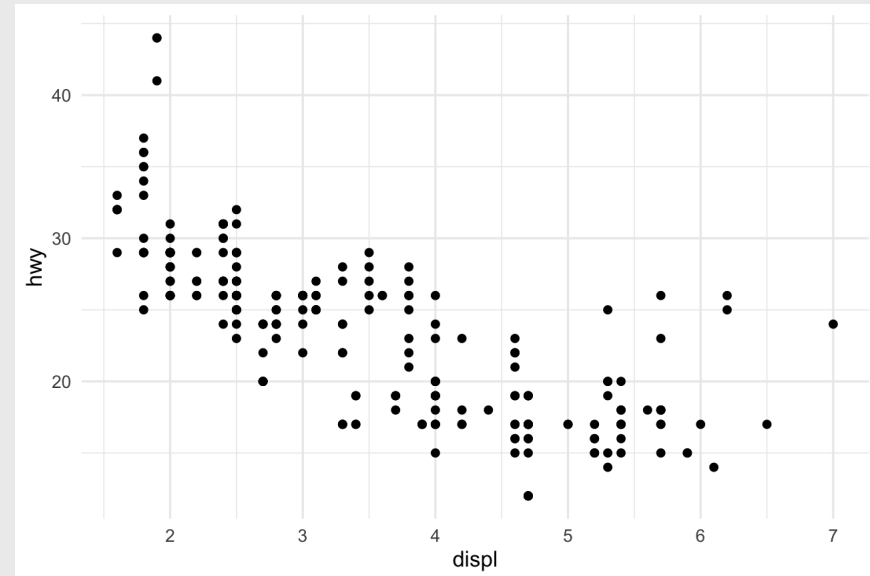
## theme\_bw()

```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  theme_bw()
```



## theme\_minimal()

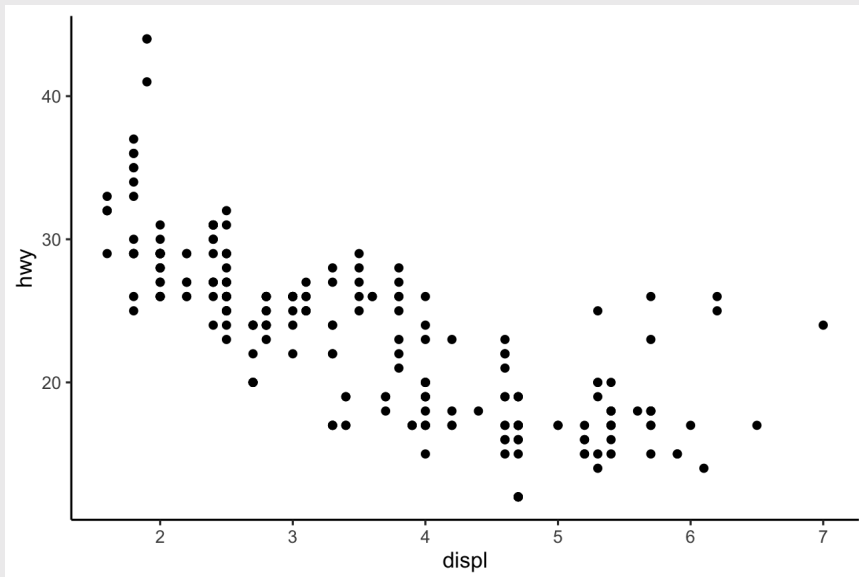
```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  theme_minimal()
```



# Common themes

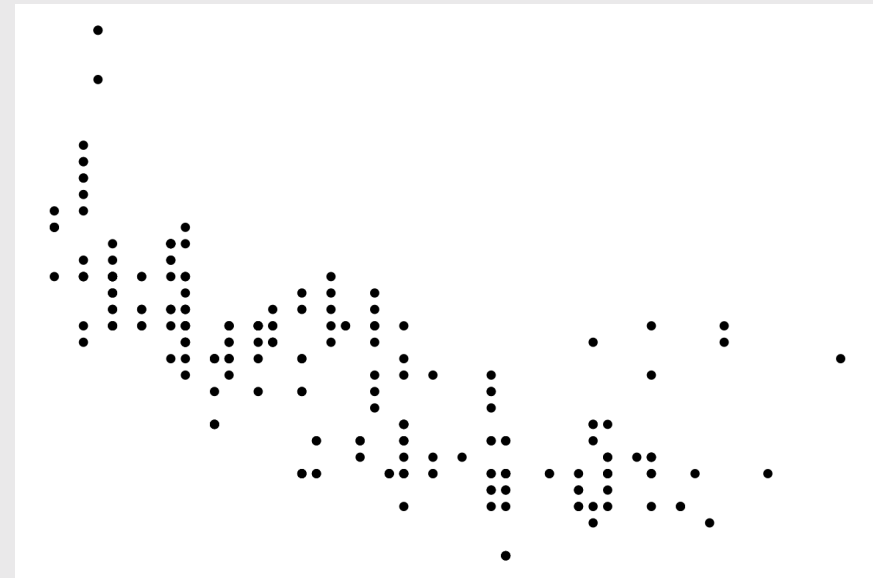
## theme\_classic()

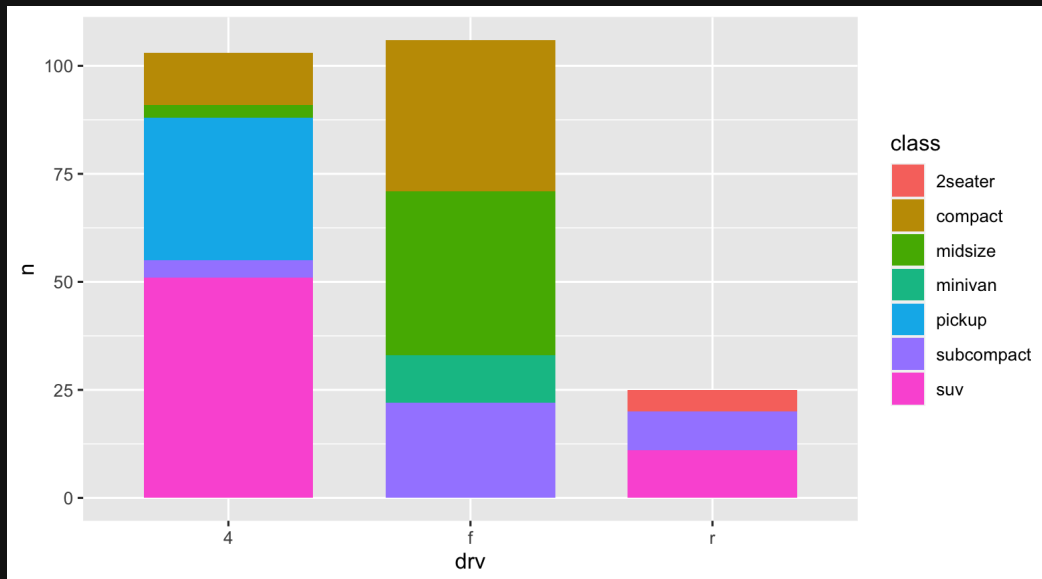
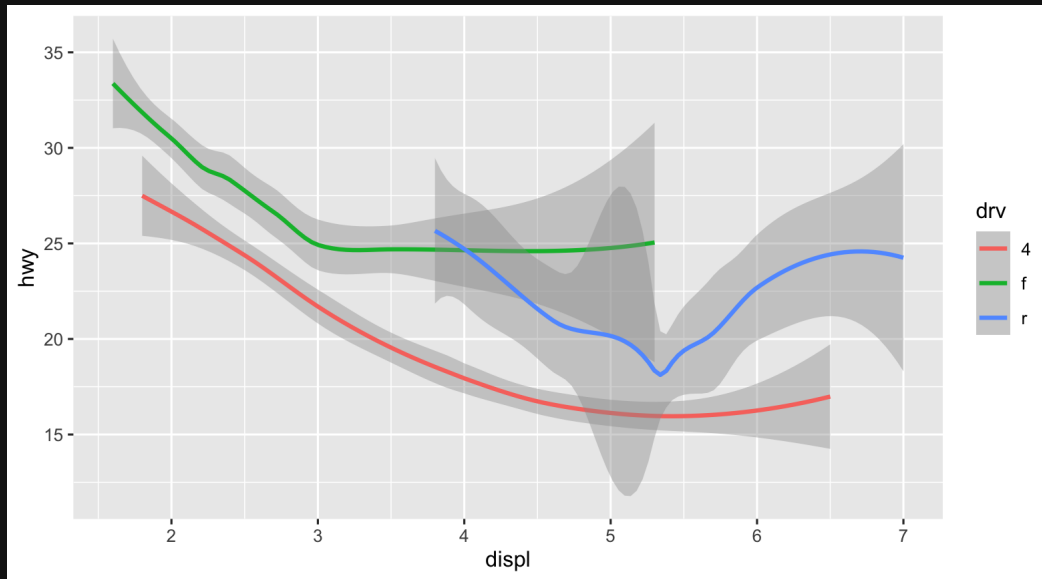
```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  theme_classic()
```



## theme\_void()

```
mpg %>%  
  ggplot(aes(x = displ, y = hwy)) +  
  geom_point() +  
  theme_void()
```



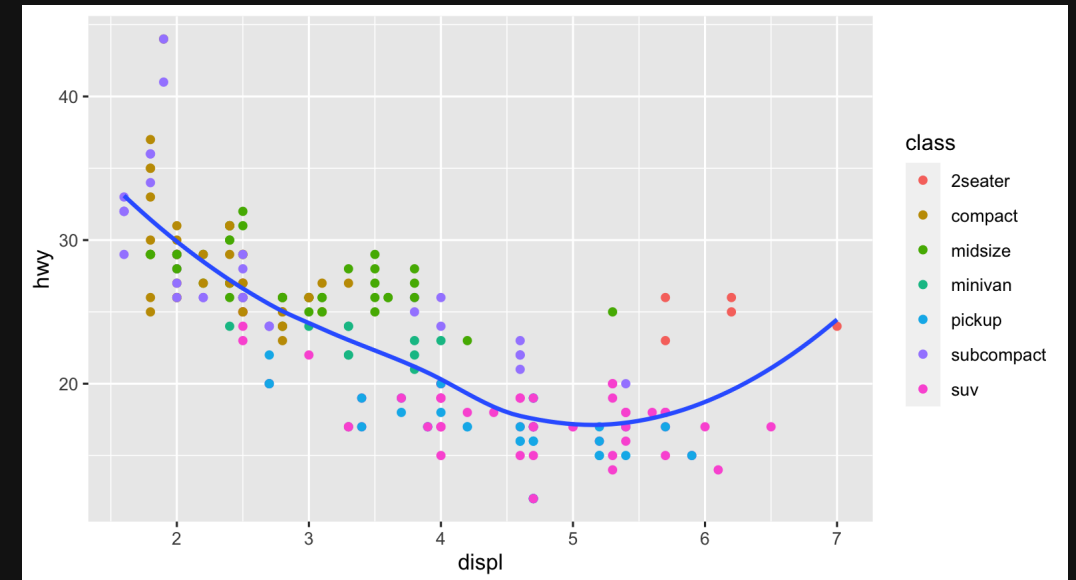


# Your turn

15:00

Open `practice.Rmd`

Use the `mpg` data frame and `ggplot` to create these charts



# Extra practice

