Topics in R: Tidyverse for Data Analysis

UCI Data Science Initiative

Dustin Pluta

2018-05-04

Overview

AM

- Intro to the Tidyverse
- Importing and wrangling data with `readr` and `tidyr`.
- Exploration and visualization with `ggplot2` and `dplyr`.

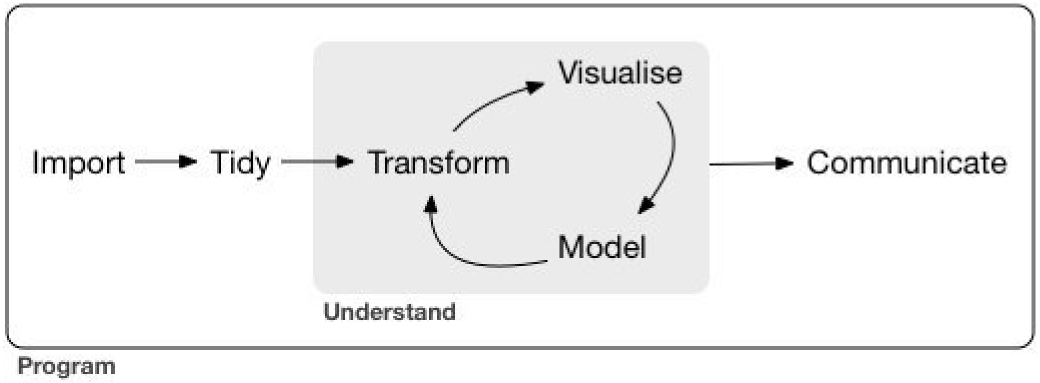
Lunch

PM

- Tidyverse in application: Data Analysis of Coronary Heart Disease Data
- Writing reports with R Markdown.

Intro to the Tidyverse

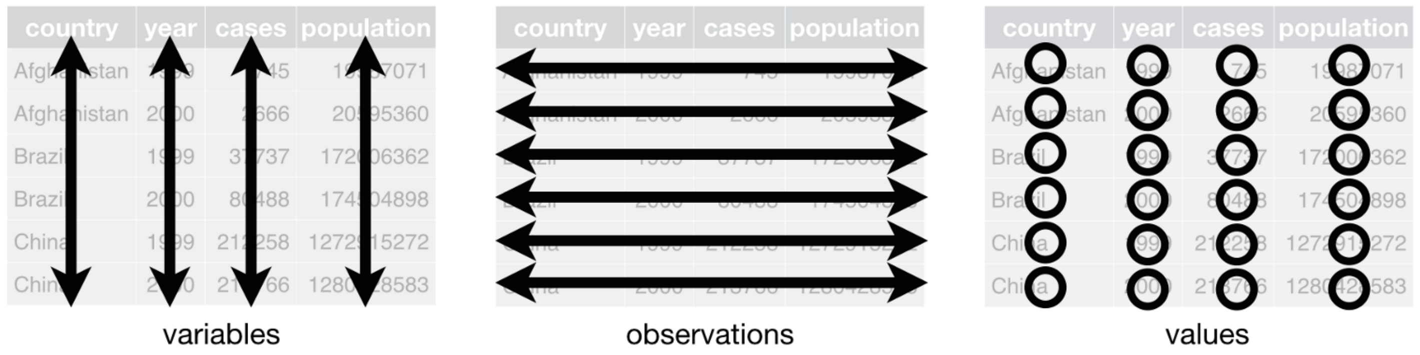
Tidy Analysis Pipeline



- *tidyverse* philosophy: collection of small, simple functions that each do one thing well
- Written by Hadley Wickham, Chief Scientist for R Studio, who also developed:
    - `ggplot2`
    - `reshape2`
    - `tidyr`
    - many others

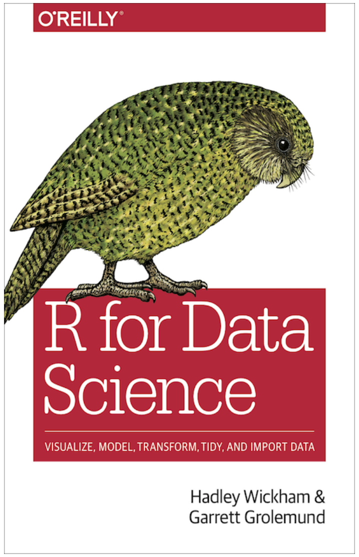
Intro to the Tidyverse

Tidy Data



Intro to the Tidyverse

Book: R for Data Science



http://r4ds.had.co.nz/

Intro to the Tidyverse

Packages

readr: import and export data

tidyr: wrangle and clean data

dplyr: slice, subset, transform, and summarize data

ggplot2: visualization

purrr: vectorized and parallel operations

RMarkdown: preparing and presenting results

Intro to Tidyverse: Getting Started

Install Packages

```r
install.packages("tidyverse")
install.packages("rmarkdown")
```

Load Tidyverse

```r
library(tidyverse)
```

Load IMDB Data

```r
imdb <- read_csv("dat/movie_metadata.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_integer(),
##   color = col_character(),
##   director_name = col_character(),
##   actor_2_name = col_character(),
##   genres = col_character(),
##   actor_1_name = col_character(),
##   movie_title = col_character(),
##   actor_3_name = col_character(),
##   plot_keywords = col_character(),
##   movie_imdb_link = col_character(),
##   language = col_character(),
##   country = col_character(),
##   content_rating = col_character(),
##   imdb_score = col_double(),
##   aspect_ratio = col_double()
## )

## See spec(...) for full column specifications.
```

IMDB Data

```r
head(imdb)
```

```
## # A tibble: 6 x 28
##   color director_name num_critic_for_… duration director_facebo…
##   <chr> <chr>                    <int>    <int>            <int>
## 1 Color James Cameron              723      178                0
## 2 Color Gore Verbins…              302      169              563
## 3 Color Sam Mendes                 602      148                0
## 4 Color Christopher …              813      164            22000
## 5 <NA>  Doug Walker                 NA       NA              131
## 6 Color Andrew Stant…              462      132              475
## # ... with 23 more variables: actor_3_facebook_likes <int>,
## #   actor_2_name <chr>, actor_1_facebook_likes <int>, gross <int>,
## #   genres <chr>, actor_1_name <chr>, movie_title <chr>,
## #   num_voted_users <int>, cast_total_facebook_likes <int>,
## #   actor_3_name <chr>, facenumber_in_poster <int>, plot_keywords <chr>,
## #   movie_imdb_link <chr>, num_user_for_reviews <int>, language <chr>,
## #   country <chr>, content_rating <chr>, budget <int>, title_year <int>,
## #   actor_2_facebook_likes <int>, imdb_score <dbl>, aspect_ratio <dbl>,
## #   movie_facebook_likes <int>
```

IMDB Data

```r
colnames(imdb)
```

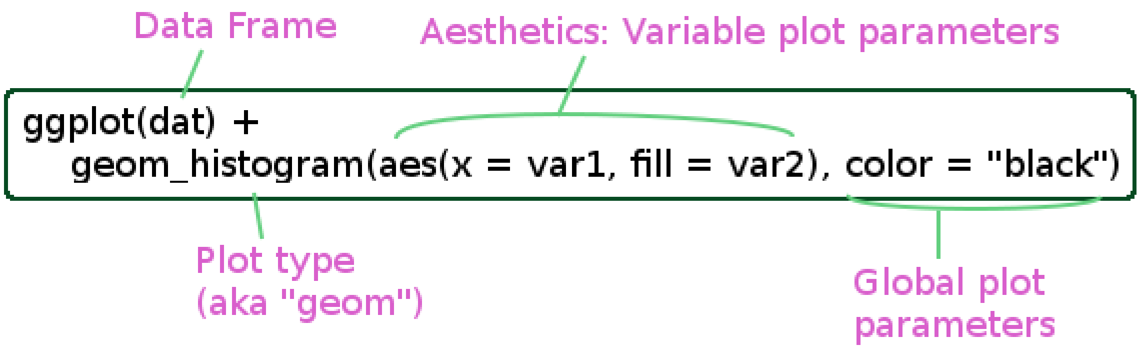
```
##  [1] "color"                     "director_name"
##  [3] "num_critic_for_reviews"    "duration"
##  [5] "director_facebook_likes"   "actor_3_facebook_likes"
##  [7] "actor_2_name"              "actor_1_facebook_likes"
##  [9] "gross"                     "genres"
## [11] "actor_1_name"              "movie_title"
## [13] "num_voted_users"           "cast_total_facebook_likes"
## [15] "actor_3_name"              "facenumber_in_poster"
## [17] "plot_keywords"             "movie_imdb_link"
## [19] "num_user_for_reviews"      "language"
## [21] "country"                   "content_rating"
## [23] "budget"                    "title_year"
## [25] "actor_2_facebook_likes"    "imdb_score"
## [27] "aspect_ratio"              "movie_facebook_likes"
```

Visualization with `ggplot2`

- `ggplot2` is a plotting package that is a nice and more modern alternative to `R` base plots
- Based on the idea of a *grammar of graphics*...
    - Think of a **plot like a sentence...**
    - **noun**: the plot data
    - **verbs**: the plot types
    - **adverbs**: the plot characteristics

Visualization with `ggplot2`

Visualization with `ggplot2`



```
ggplot(dat) +
    geom_histogram(aes(x = var1, fill = var2), color = "black")
```

Visualization with `ggplot2`

List of `geoms`

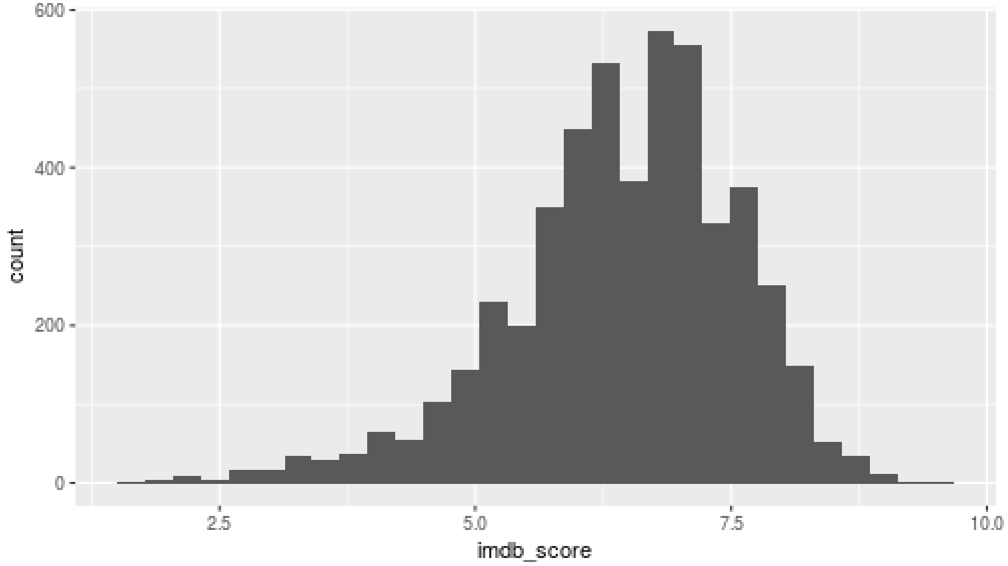
- `geom_histogram`
- `geom_density`
- `geom_point`
- `geom_line`
- `geom_boxplot`
- … many others

Visualization with `ggplot2`

```r
library(ggplot2)
```

```r
ggplot(imdb) +
  geom_histogram(aes(x = imdb_score))
```

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



Visualization with `ggplot2`

Some Aesthetics

x: horizontal position

y: vertical position

alpha: transparency

color: border color

fill: interior color

group: grouping variable

linetype

size

Visualization with `ggplot2`

Different geoms have different aesthetics

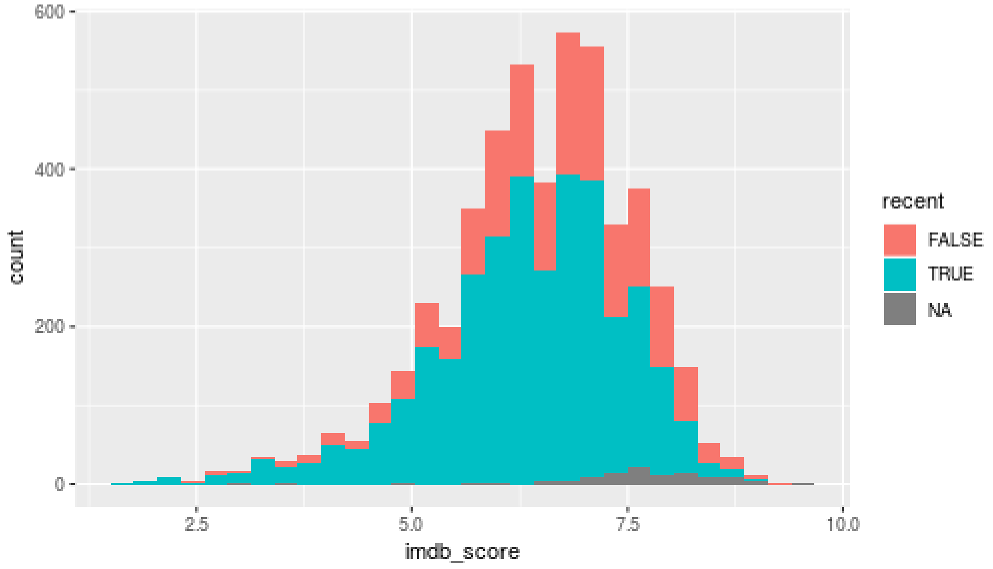
Refer to the documentation to see which aesthetics are supported for a geom

```
?geom_histogram
```

Visualization with `ggplot2`

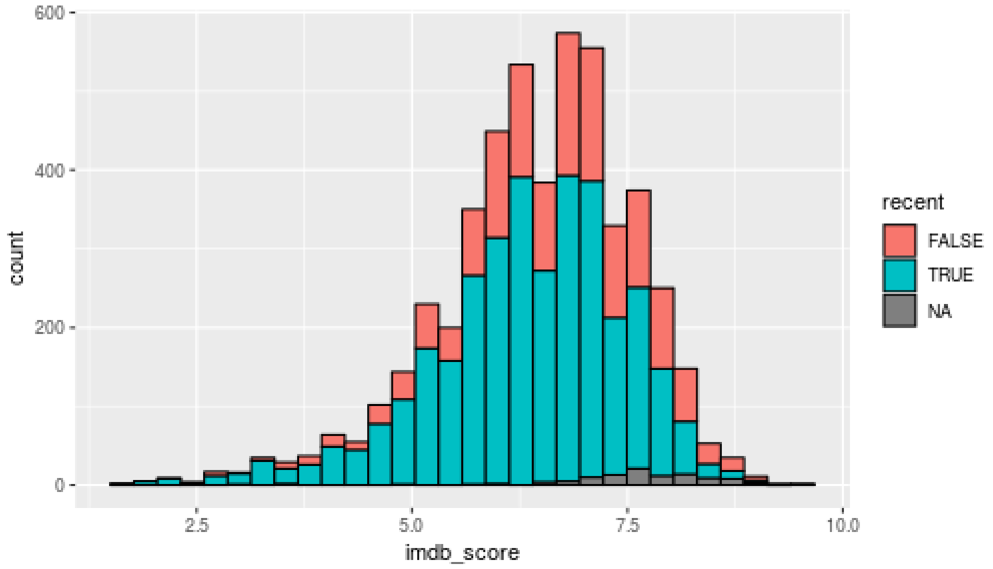
```r
imdb$recent <- imdb$title_year > 2000
ggplot(imdb) +
  geom_histogram(aes(x = imdb_score, fill = recent))
```

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



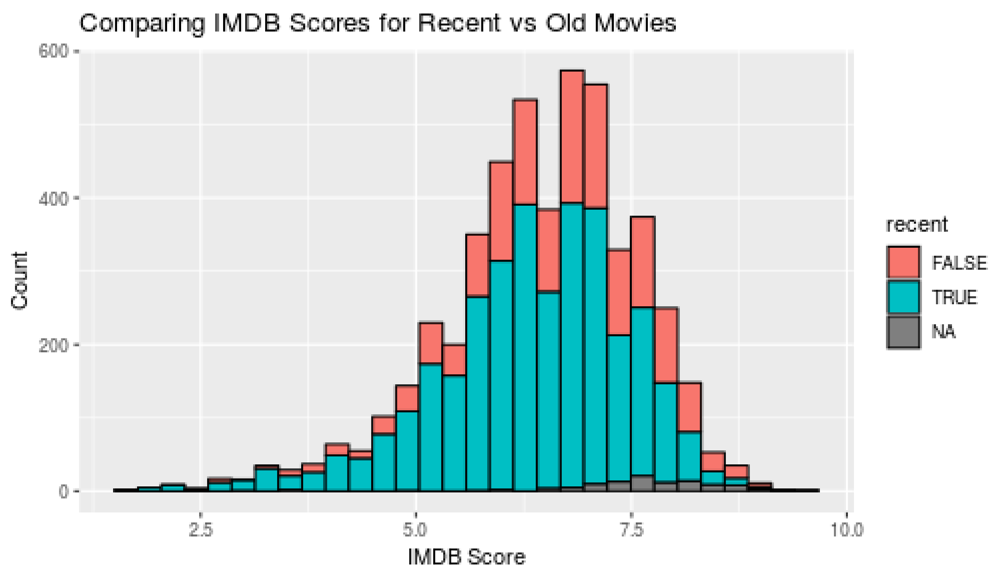
Visualization with `ggplot2`

```r
imdb$recent <- imdb$title_year > 2000
ggplot(imdb) +
  geom_histogram(aes(x = imdb_score, fill = recent),
                 color = "black", bins = 30)
```



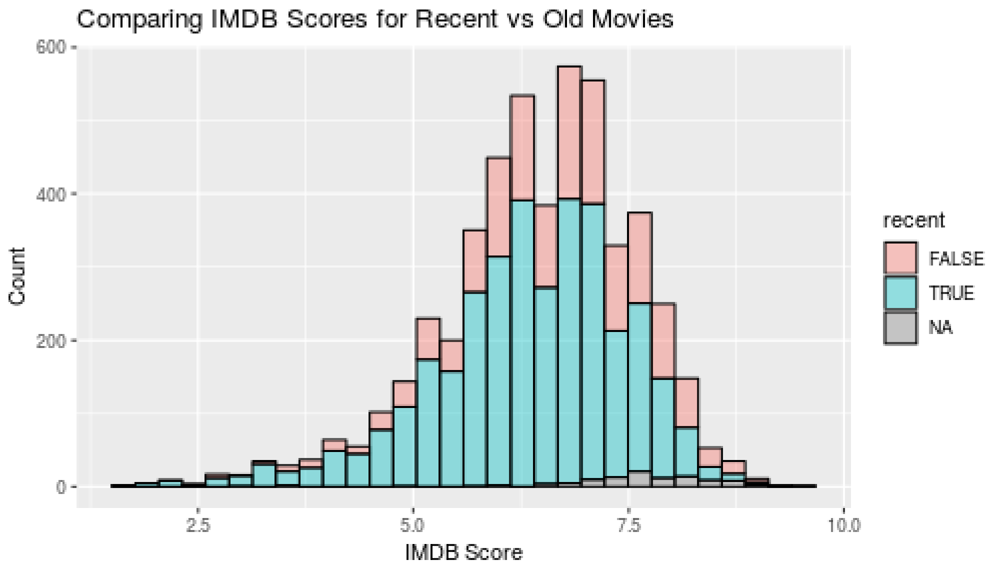
Visualization with `ggplot2`

```r
ggplot(imdb) +
  geom_histogram(aes(x = imdb_score, fill = recent),
                 color = "black", bins = 30) +
  xlab("IMDB Score") +
  ylab("Count") +
  ggtitle("Comparing IMDB Scores for Recent vs Old Movies")
```



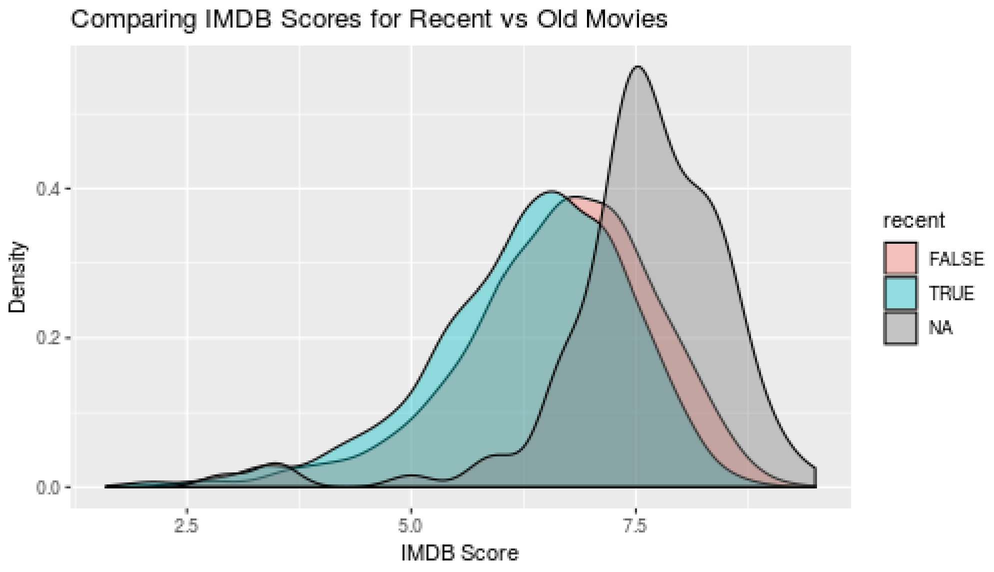
Visualization with `ggplot2`

```r
ggplot(imdb) +
  geom_histogram(aes(x = imdb_score, fill = recent),
                 color = "black", bins = 30, alpha = 0.4) +
  xlab("IMDB Score") +
  ylab("Count") +
  ggtitle("Comparing IMDB Scores for Recent vs Old Movies")
```



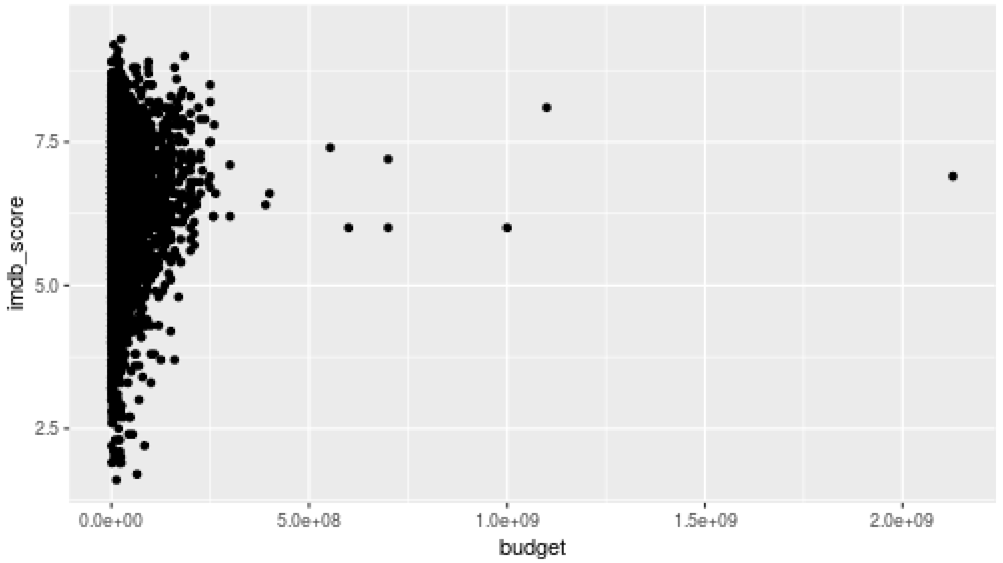
Visualization with `ggplot2`

```r
ggplot(imdb) +
  geom_density(aes(x = imdb_score, fill = recent),
               color = "black", alpha = 0.4) +
  xlab("IMDB Score") +
  ylab("Density") +
  ggtitle("Comparing IMDB Scores for Recent vs Old Movies")
```



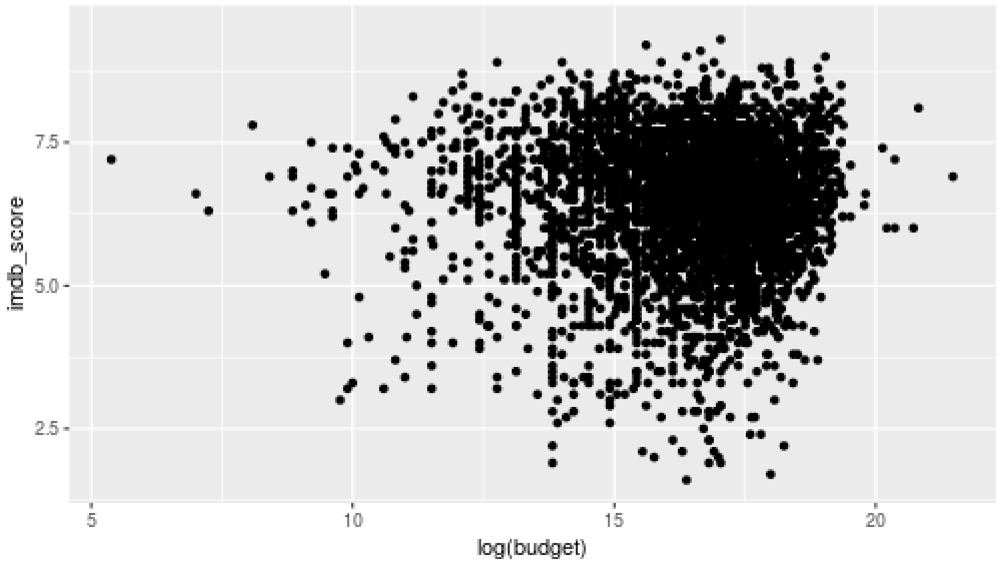
Visualization with `ggplot2`

```
ggplot(imdb) +
  geom_point(aes(x = budget, y = imdb_score))
```



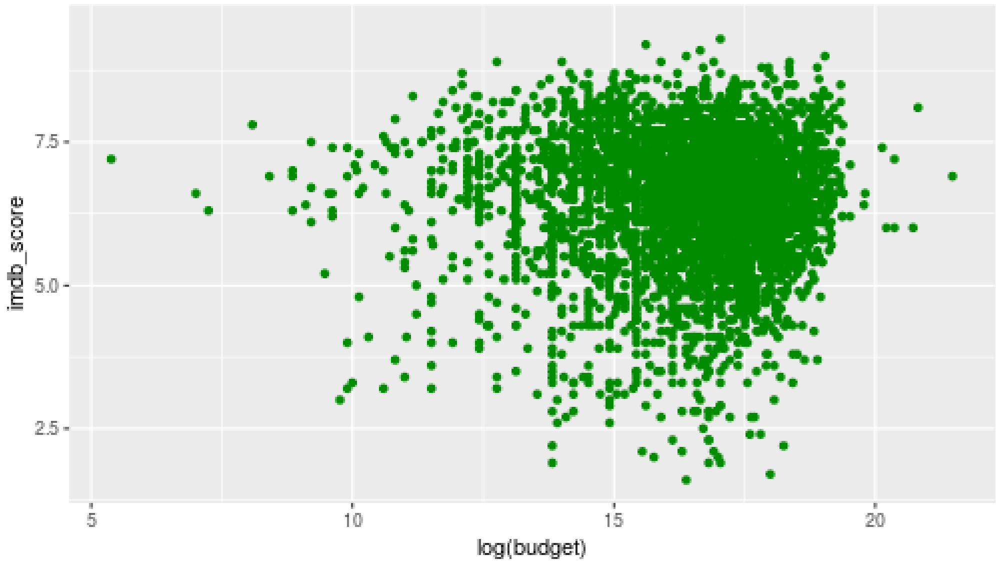
Visualization with `ggplot2`

```
ggplot(imdb) +
  geom_point(aes(x = log(budget), y = imdb_score))
```



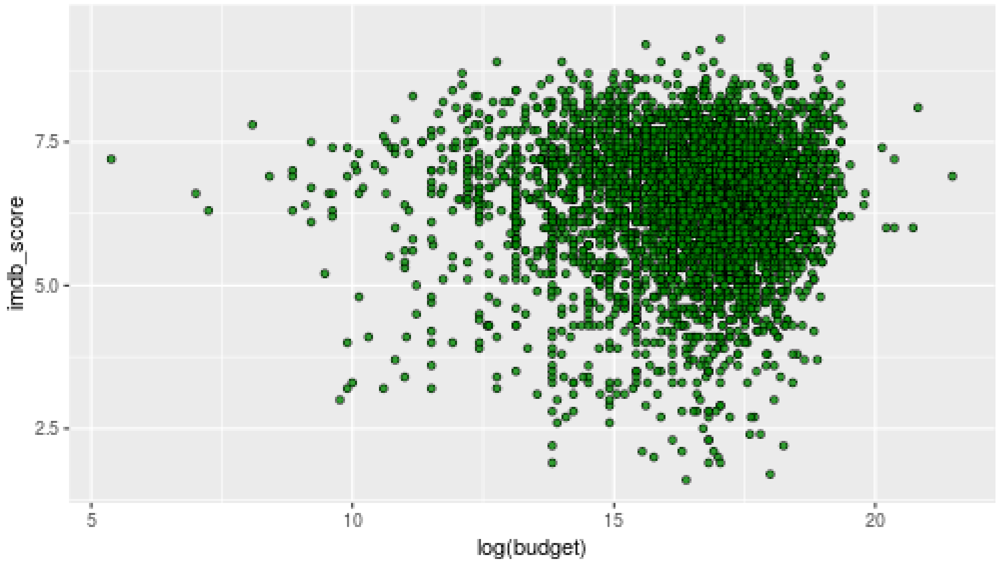
Visualization with `ggplot2`

```
ggplot(imdb) +
  geom_point(aes(x = log(budget), y = imdb_score), color = "green4")
```



Visualization with `ggplot2`

```
ggplot(imdb) +
  geom_point(aes(x = log(budget), y = imdb_score),
             pch = 21, fill = "green4",
             color = "black", alpha = 0.8)
```



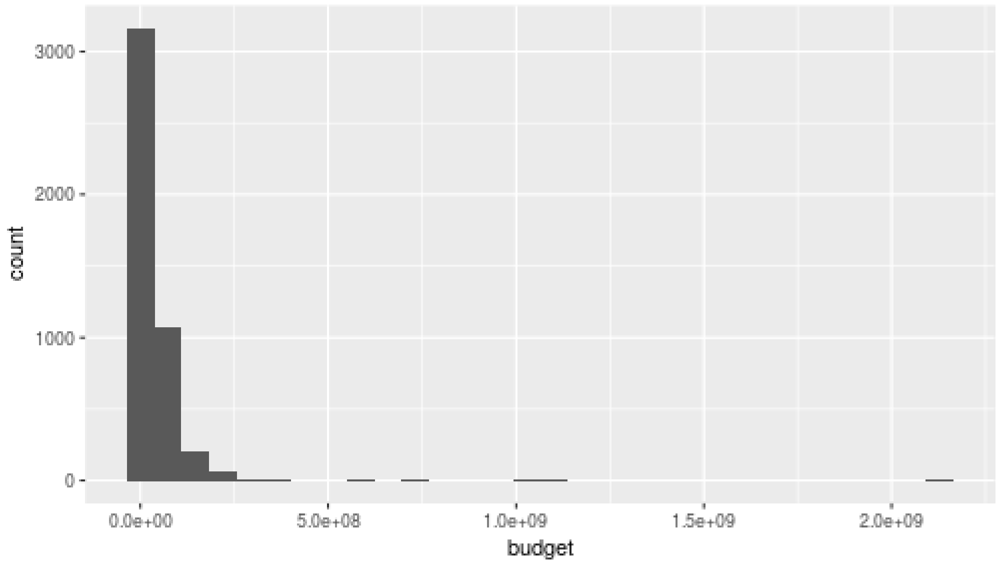
Visualization Exercises

1. Plot a histogram of `budget` and compare it to a histogram of `log(budget)`.
2. Add some color, change the title and axis labels for the `log(budget)` histogram.
3. Make a new variable `recent` to indicate if a movie is more recent than 2000 using `imdb$recent <- imdb$title_year > 2000`, then plot a histogram of `log(budget)` grouped by `recent`.
4. Create a scatterplot of `imdb_score` by `log(budget)` and colored by `recent`.
5. Create a boxplot of `imdb_score` grouped by `recent`, using `geom_boxplot`.

Visualization Exercise # 1

```
ggplot(imdb) +
  geom_histogram(aes(x = budget))
```

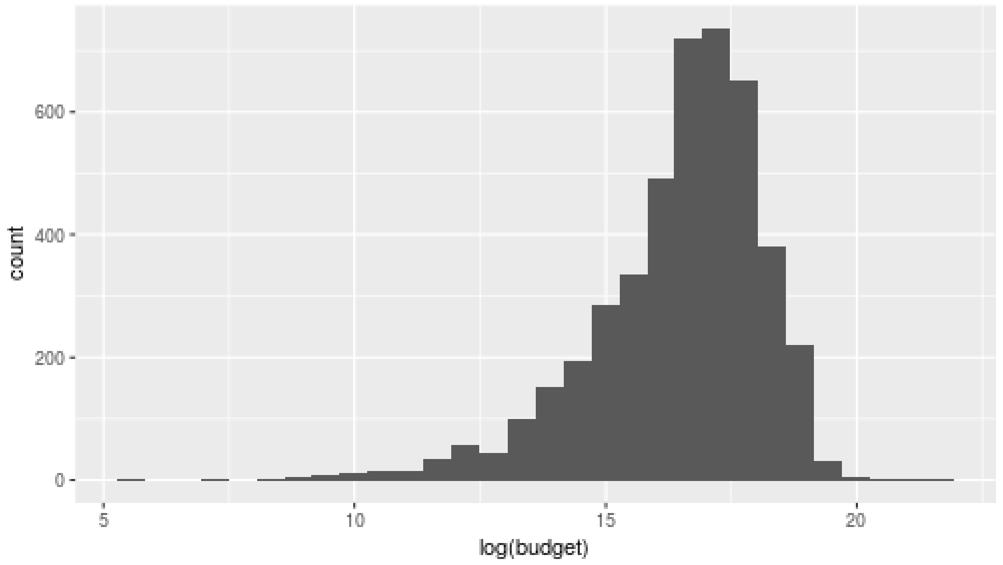
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Visualization Exercise # 1

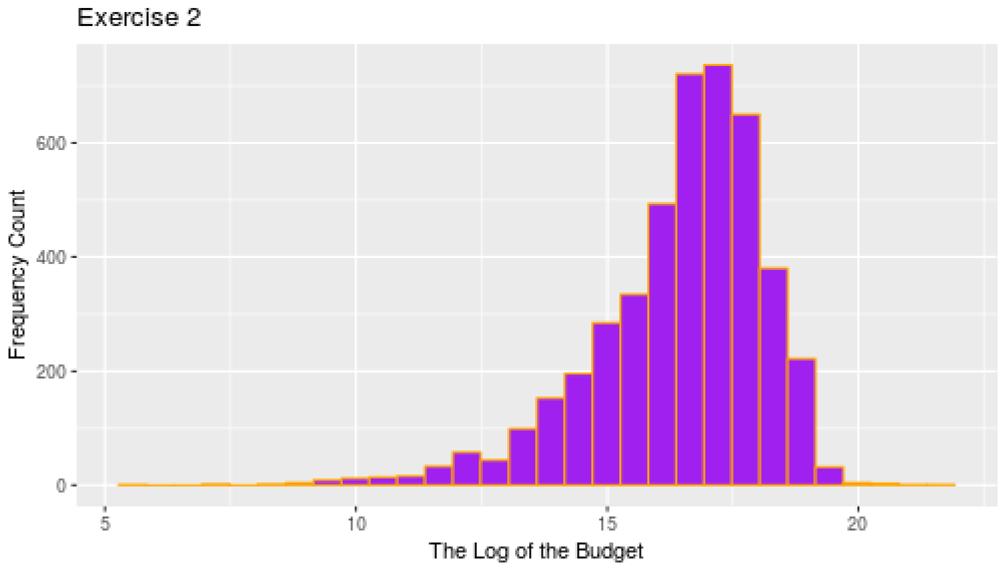
```r
ggplot(imdb) +
  geom_histogram(aes(x = log(budget)))
```

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



Visualization Exercise # 2

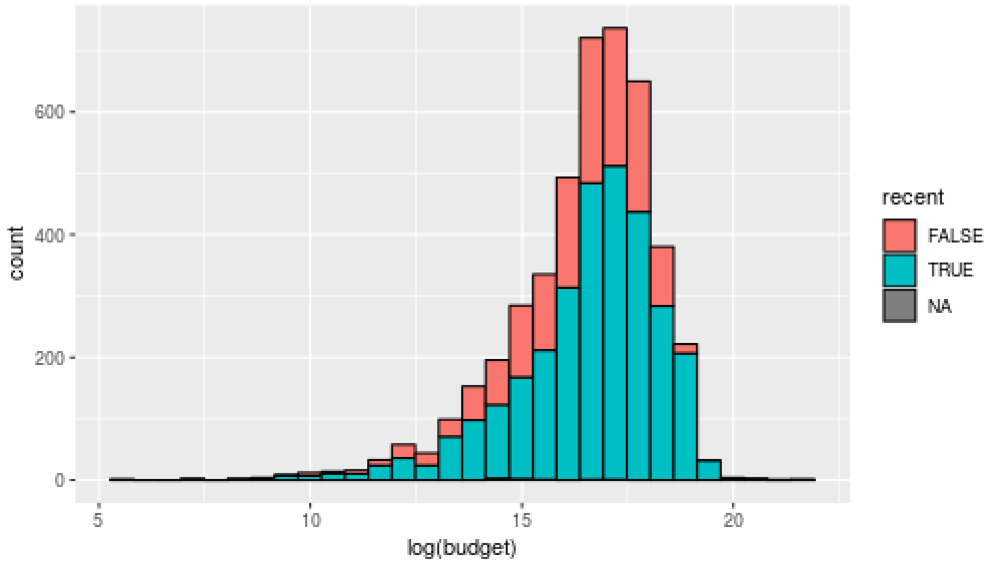
```
ggplot(imdb) +
  geom_histogram(aes(x = log(budget)),
                 color = "orange", fill = "purple", bins = 30) +
  ggtitle("Exercise 2") +
  xlab("The Log of the Budget") +
  ylab("Frequency Count")
```



Visualization Exercise # 3

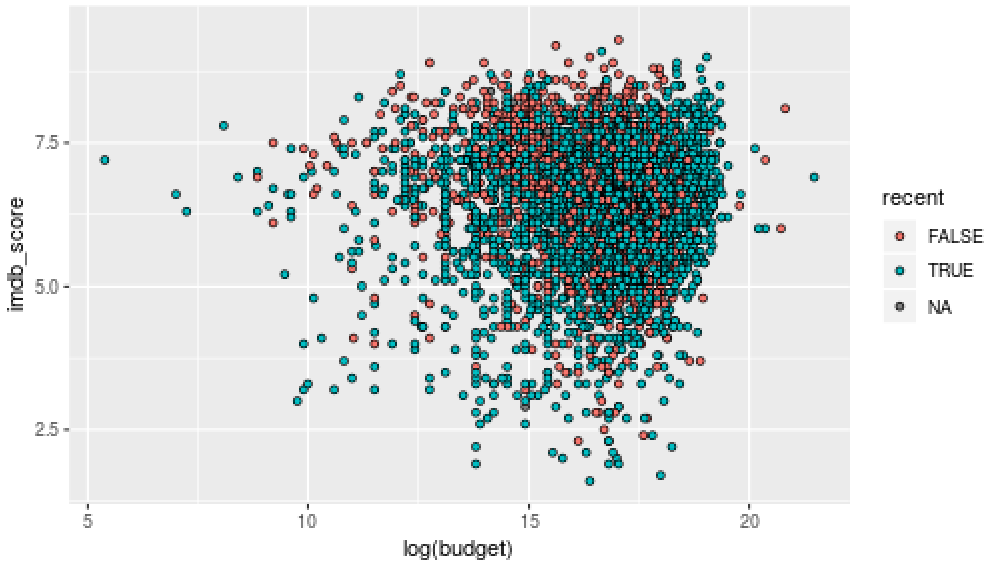
```r
imdb$recent <- imdb$title_year > 2000
ggplot(imdb) +
  geom_histogram(aes(x = log(budget), fill = recent),
                 color = "black")
```

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



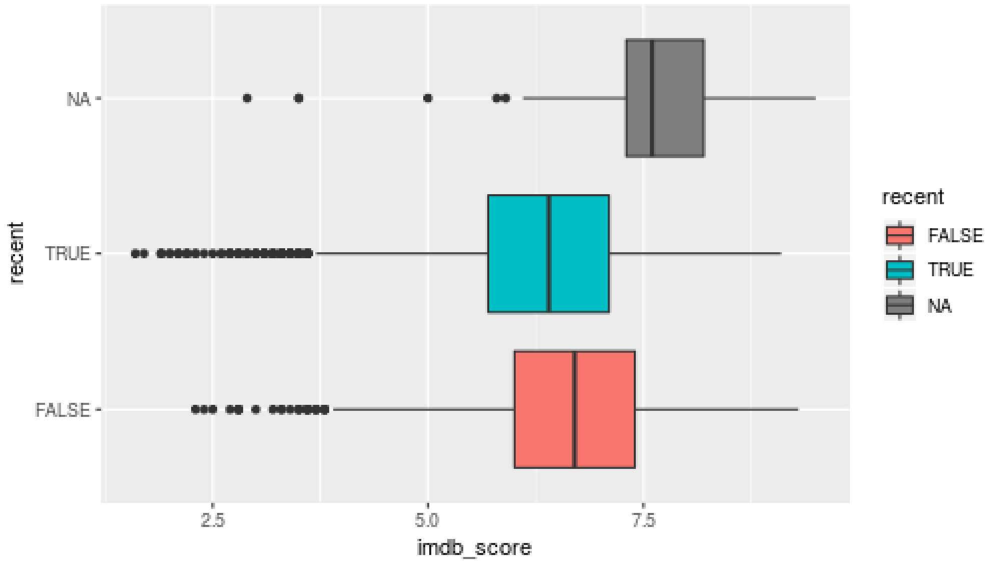
Visualization Exercise # 4

```r
ggplot(imdb) +
  geom_point(aes(x = log(budget), y = imdb_score, fill = recent),
             pch = 21, color = "black")
```



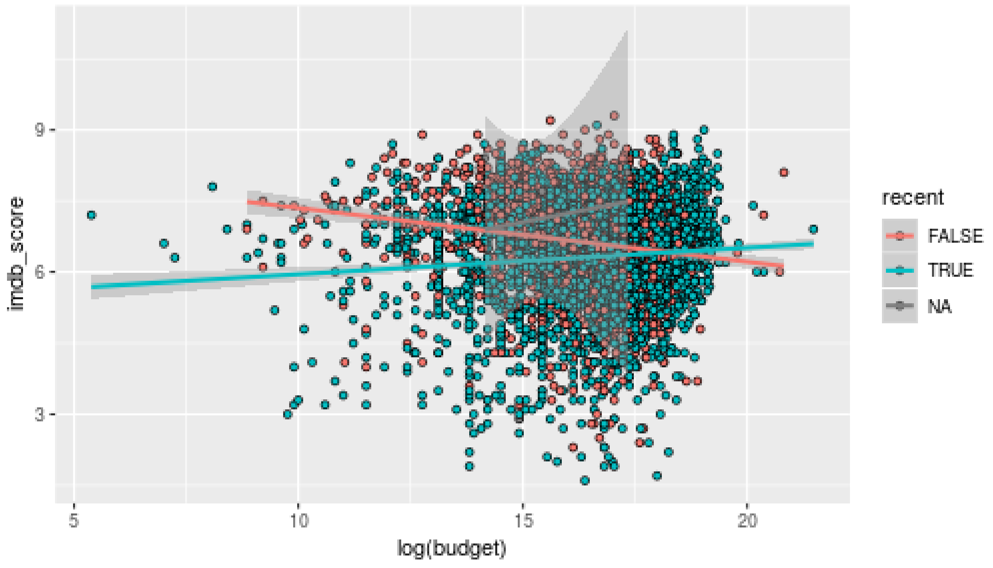
Visualization Exercises # 5

```
ggplot(imdb) +
  geom_boxplot(aes(y = imdb_score, x = recent, fill = recent)) +
  coord_flip()
```



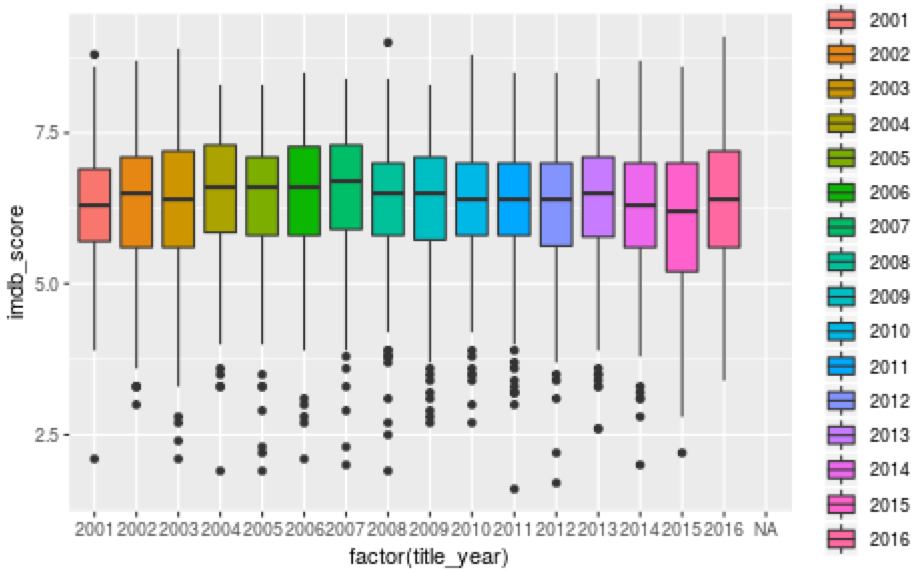
```
ggplot(imdb) +
  geom_qq(aes(sample = imdb_score)) +
  geom_qq_line(aes(sample = imdb_score))
```

```r
ggplot(imdb) +
  geom_point(aes(x = log(budget), y = imdb_score, fill = recent),
             pch = 21, color = "black") +
  geom_smooth(aes(x = log(budget), y = imdb_score, color = recent),
              method = "lm")
```



Visualization with `ggplot2`

```
ggplot(imdb[imdb$recent, ]) +
  geom_boxplot(aes(y = imdb_score, fill = factor(title_year),
                   x = factor(title_year)))
```



Data Wrangling with `dplyr`

Data Wrangling with `dplyr`

- `dplyr` is a package designed for easy and efficient data manipulation

Key Functions

- `filter`: select subset of rows (observations)
- `select`: select subset of columns (variables)
- `mutate`: transform variables in a data set
- `arrange`: reorder rows
- `summarise`: collapses a data frame into a single row
- `group_by`

Data Wrangling with `dplyr`

- Let's try some `dplyr` functions with the `iris` data set:

```
# Print species means of Sepal Width
data(iris)
iris <-  filter(iris, Species!="setosa")
iris <- select(iris, c(Sepal.Width, Species))
iris <- group_by(iris, Species)
species_means <- summarise(iris, mean(Sepal.Width))
print(species_means)
```

```
## # A tibble: 2 x 2
##   Species    `mean(Sepal.Width)`
##   <fct>                    <dbl>
## 1 versicolor                2.77
## 2 virginica                 2.97
```

Data Wrangling with `dplyr`

Introducing the Pipe: `%>%`

- `dplyr` (and much of the *tidyverse*) is designed around the use of the pipe operator `%>%`
- The pipe operator `%>%` allows you to chain operations on a data set together without having to create specific intermediate objects
- When using `%>%`, the first argument to a function is taken as the output of the previous step in the chain

Data Wrangling with `dplyr`

- For example, the following is equivalent to the previous code:

```r
# Prints species means, does not save anything
# Original data.frame iris is unaffected
data(iris)
iris %>% filter(Species!="setosa") %>%
    select(c(Sepal.Width, Species)) %>%
    group_by(Species) %>%
    summarise(mean(Sepal.Width))
```

```
## # A tibble: 2 x 2
##   Species    `mean(Sepal.Width)`
##   <fct>                    <dbl>
## 1 versicolor                2.77
## 2 virginica                 2.97
```

Data Wrangling with `dplyr`

```r
# To save the results instead
species_means <- iris %>%
    filter(Species!="setosa") %>%
    select(c(Sepal.Width, Species)) %>%
    group_by(Species) %>%
    summarise(mean(Sepal.Width))
```

```r
species_means
```

```
## # A tibble: 2 x 2
##   Species    `mean(Sepal.Width)`
##   <fct>                    <dbl>
## 1 versicolor                2.77
## 2 virginica                 2.97
```

`dplyr` Exercises

1. Use `dplyr` to calculate the mean Sepal Width of the virginica species.

2. `summarise` can summarise multiple variables simultaneously, applying a (possibly different) function to each variable.
   Adapt the code below to find the minimum, median, maximum, and standard deviation of the Sepal.Width for the virginica species.

3. `group_by()` makes `summarise` even more useful by allowing you to summarise values across groups of a category simultaneously.
   Using `group_by`, adapt your code from the previous problem to produce the summary values for each species.

Modify this code for problems 2 and 3:

```
data(iris)
iris %>% summarise(mean_sepal_width = mean(Sepal.Width),
                   min_sepal_width = min(Sepal.Width))
```

dplyr Exercises

Solution

- (1) Use dplyr to calculate the mean Sepal Width of the virginica species.

```
data(iris)

iris %>%
    filter(Species == "virginica") %>%
    summarise(mean_sepal_width = mean(Sepal.Width))
```

```
##   mean_sepal_width
## 1            2.974
```

dplyr Exercises

Solution

- (2) summarise can summarise multiple variables simultaneously, applying a (possibly different) function to each variable.
  Adapt the code below to find the minimum, median, maximum, and standard deviation of the Sepal.Width for the virginica species.

```
data(iris)
iris %>%
    filter(Species == "virginica") %>%
    summarise(min_sepal_width = min(Sepal.Width),
              med = median(Sepal.Width), maximum = max(Sepal.Width),
              stdev = sd(Sepal.Width))
```

```
##   min_sepal_width med maximum     stdev
## 1             2.2   3     3.8 0.3224966
```

`dplyr` Exercises

Solution

- (3) `group_by()` makes `summarise` even more useful by allowing you to summarise values across groups of a category simultaneously.
  Using `group_by`, adapt your code from the previous problem to produce the summary values for each species.

```
data(iris)
iris %>%
    group_by(Species) %>%
    summarise(min_sepal_width = min(Sepal.Width),
              med = median(Sepal.Width), maximum = max(Sepal.Width),
              stdev = sd(Sepal.Width))
```

```
## # A tibble: 3 x 5
##   Species    min_sepal_width   med maximum stdev
##   <fct>                <dbl> <dbl>   <dbl> <dbl>
## 1 setosa                 2.3   3.4     4.4 0.379
## 2 versicolor             2     2.8     3.4 0.314
## 3 virginica              2.2   3       3.8 0.322
```

Back to the Movies

Exploring the IMDB Data

How many movies for each actor in the dataset?

```
imdb %>%
  group_by(actor_1_name) %>%
  summarize(n())
```

Exploring the IMDB Data

How many movies for each actor in the dataset?

```
imdb %>%
  group_by(actor_1_name) %>%
  summarize(n())
```

```
## # A tibble: 2,098 x 2
##    actor_1_name      `n()`
##    <chr>             <int>
##  1 50 Cent               1
##  2 Aaliyah               1
##  3 Aasif Mandvi          1
##  4 Abbie Cornish         3
##  5 Abhishek Bachchan     2
##  6 Abigail Evans         1
##  7 Abigail Spencer       1
##  8 Adam Arkin            2
##  9 Adam Baldwin          4
## 10 Adam Garcia           3
## # ... with 2,088 more rows
```

Exploring the IMDB Data

How many movies for each actor in the dataset?

Arranged by decreasing number of movies

```
imdb %>%
  group_by(actor_1_name) %>%
  summarize(n_movies = n()) %>%
  arrange(desc(n_movies))
```

```
## # A tibble: 2,098 x 2
##    actor_1_name      n_movies
##    <chr>                <int>
##  1 Robert De Niro          49
##  2 Johnny Depp             40
##  3 Nicolas Cage            32
##  4 J.K. Simmons            31
##  5 Bruce Willis            30
##  6 Denzel Washington       30
##  7 Matt Damon              30
##  8 Liam Neeson             29
##  9 Harrison Ford           27
## 10 Robin Williams          27
```

Exploring the IMDB Data

How many movies for each actor in the dataset?

Arranged by decreasing mean IMDB score

```
imdb %>%
  group_by(actor_1_name) %>%
  summarize(mean_imdb_score = mean(imdb_score)) %>%
  arrange(desc(mean_imdb_score))
```

```
## # A tibble: 2,098 x 2
##    actor_1_name       mean_imdb_score
##    <chr>                        <dbl>
##  1 Krystyna Janda                 9.1
##  2 Jack Warden                    8.9
##  3 Rob McElhenney                 8.8
##  4 Abigail Evans                  8.7
##  5 Elina Abai Kyzy                8.7
##  6 Jackie Gleason                 8.7
##  7 Kimberley Crossman             8.7
##  8 Maria Pia Calzone              8.7
##  9 Takashi Shimura                8.7
## 10 Bunta Sugawara                 8.6
```

Exploring the IMDB Data

Considering actors with more than 5 movies, list top 10 actors with highest mean IMDB scores in decreasing order.

```
imdb %>%
  group_by(actor_1_name) %>%
  summarize(mean_imdb_score = mean(imdb_score), n_movies = n()) %>%
  filter(n_movies > 5) %>%
  top_n(10, mean_imdb_score) %>%
  arrange(desc(mean_imdb_score))
```

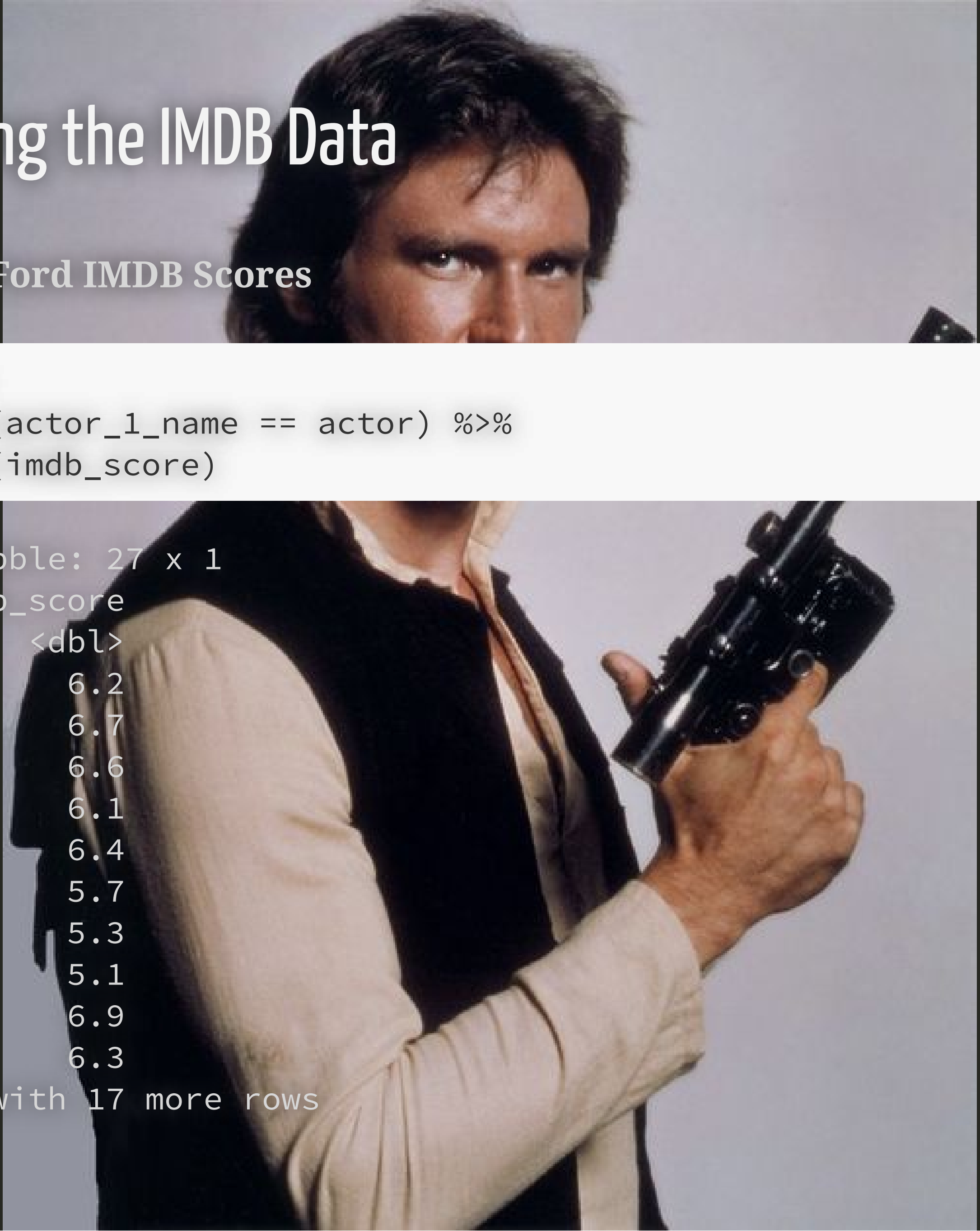
```
## # A tibble: 10 x 3
##    actor_1_name             mean_imdb_score n_movies
##    <chr>                              <dbl>    <int>
##  1 Leonardo DiCaprio                   7.50       21
##  2 Tom Hanks                           7.42       24
##  3 Clint Eastwood                      7.34       16
##  4 Tom Hardy                           7.31       11
##  5 Alan Rickman                        7.29        8
##  6 Benedict Cumberbatch                7.29        7
##  7 Philip Seymour Hoffman              7.24       20
##  8 Toby Jones                          7.22        6
##  9 Minnie Driver                       7.21        7
```

Exploring the IMDB Data

How many movie entries does Harrison Ford have?

```r
actor <- "Harrison Ford"
filter(imdb, actor_1_name == actor) %>% nrow
```

```
## [1] 27
```



Exploring the IMDB Data

Harrison Ford IMDB Scores

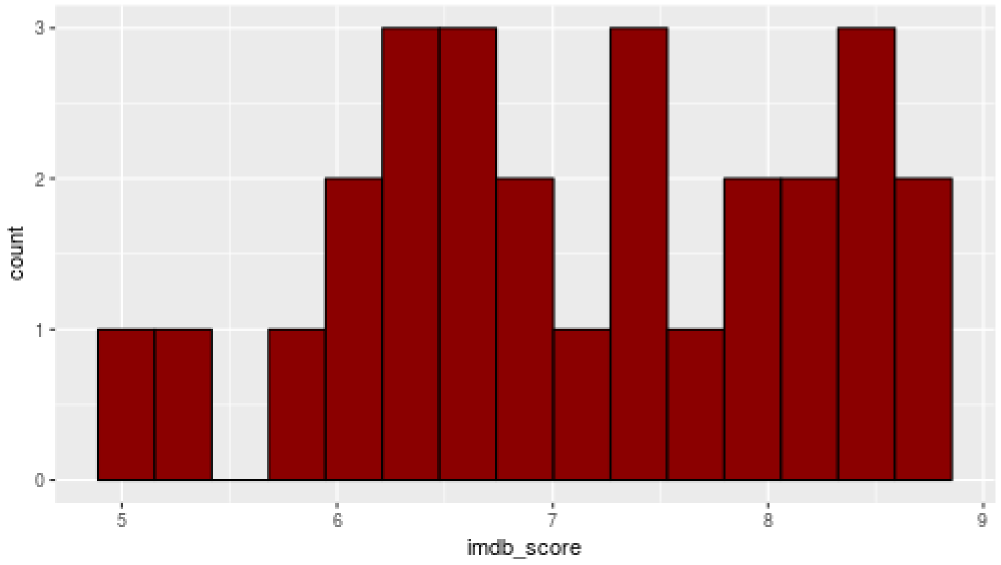
```
imdb %>%
  filter(actor_1_name == actor) %>%
  select(imdb_score)
```

```
## # A tibble: 27 x 1
##    imdb_score
##         <dbl>
##  1        6.2
##  2        6.7
##  3        6.6
##  4        6.1
##  5        6.4
##  6        5.7
##  7        5.3
##  8        5.1
##  9        6.9
## 10        6.3
## # ... with 17 more rows
```

Exploring the IMDB Data

Histogram of Harrison Ford IMDB Scores

```
ggplot(imdb %>% filter(actor_1_name == actor)) +
  geom_histogram(aes(x = imdb_score),
                 fill = "darkred", color = "black", bins = 15)
```



Exploring the IMDB Data

Harrison By Genre

```
imdb %>%
  filter(actor_1_name %in% actor) %>%
  group_by(genres) %>%
  summarize(mean_score = mean(imdb_score), n_movies = n())
```

```
## # A tibble: 22 x 3
##    genres                                         mean_score n_movies
##    <chr>                                               <dbl>    <int>
##  1 Action|Adventure                                     8.05        2
##  2 Action|Adventure|Comedy|Romance                      5.7         1
##  3 Action|Adventure|Crime|Drama|Mystery|Thriller        7.8         1
##  4 Action|Adventure|Drama|Thriller                      6.4         1
##  5 Action|Adventure|Fantasy                             7.25        2
##  6 Action|Adventure|Fantasy|Sci-Fi                      8.63        3
##  7 Action|Comedy|Crime|Thriller                         5.3         1
##  8 Action|Crime|Drama|Thriller                          6.5         2
##  9 Action|Drama|War                                     6.3         1
## 10 Action|Sci-Fi                                        6.7         1
## # ... with 12 more rows
```

Exploring the IMDB Data

Harrison's Action Movies

```
imdb %>%
  filter(actor_1_name %in% actor) %>%
  transmute(action = str_detect(genres, "Action"))
```

```
## # A tibble: 27 x 1
##    action
##    <lgl>
##  1 TRUE
##  2 TRUE
##  3 FALSE
##  4 TRUE
##  5 TRUE
##  6 TRUE
##  7 TRUE
##  8 FALSE
##  9 TRUE
## 10 FALSE
## # ... with 17 more rows
```

Exploring the IMDB Data

Harrison's Action Movies

```r
imdb %>%
  filter(actor_1_name %in% actor) %>%
  mutate(action = str_detect(genres, "Action")) %>%
  group_by(action) %>%
  summarize(mean(imdb_score))
```

```
## # A tibble: 2 x 2
##   action `mean(imdb_score)`
##   <lgl>               <dbl>
## 1 FALSE                7.15
## 2 TRUE                 7.16
```

Tidyverse in Application

Tidyverse for Data Analysis

Need additional packages:

- `boot`
- `factoextra`
- `FactoMineR`
- `qwraps2`
- `broom`

Goals

- Practice using the tidyverse for analysis of a real world data set
- Will use data from a study of Coronary Heart Disease (CHD) in older adults
- Interest in determining factors associated with higher incidence rate of CHD

Coronary Heart Disease Data

- Covariates collected include:
  - `choltot`: Total cholesterol
  - `incchd`: Binary indicator of CHD
  - `hdl`: High-density lipoproteins
  - `alcoh`: Level of alcohol consumption
  - `trig`: Triglyceride level
  - `age`: Age in years
  - `diabetes`: Indicator of diabetes (3 levels: 1/2/3 for No, Type I, Type II)
  - `diabp`: Diastolic blood pressure
  - `sysbp`: Systolic blood pressure
  - `height`
  - `weight`
  - `smoke`: Binary indicator
  - `racebw`: Binary indactor of race white/black (0/1)
  - `gender`: Binary indicator of male/female (0/1)

CHD Data

```r
dat <- read_csv("dat/chddata_dsi.csv")
```

```
## Parsed with column specification:
## cols(
##   X1 = col_integer(),
##   idno = col_integer(),
##   choltot = col_double(),
##   incchd = col_integer(),
##   hdl = col_integer(),
##   alcoh = col_double(),
##   bmi = col_double(),
##   trig = col_integer(),
##   gender = col_integer(),
##   age = col_integer(),
##   diabetes = col_integer(),
##   diabp = col_double(),
##   sysbp = col_double(),
##   height = col_integer(),
##   weight = col_integer(),
##   smoke = col_integer(),
##   racebw = col_integer()
## )
```

CHD Data

```r
dat
```

```
## # A tibble: 3,425 x 17
##        X1    idno choltot incchd   hdl alcoh   bmi  trig gender   age
##     <int>   <int>   <dbl>  <int> <int> <dbl> <dbl> <int>  <int> <int>
##  1      1 3.01e6    224.       0    37  0     36.5   168      1    65
##  2      2 3.02e6    259.       0    50  0.06  31.0   136      0    65
##  3      3 3.03e6    205.       0    51  4     33.3   114      0    65
##  4      4 3.04e6    174.       1    34  0     31.2   332      1    65
##  5      5 3.50e6    191.       1    47  0     30.9   139      0    65
##  6      6 3.50e6    201.       0    83  0     24.8    91      0    65
##  7      7 3.50e6    161.       0    45  8.02  21.7   154      0    65
##  8      8 3.50e6    231.       0    33  0     26.6   304      1    65
##  9      9 3.50e6    229.       0    48  0     27.2   162      0    65
## 10     10 3.50e6    248.       0    47  0     22.6   101      0    65
## # ... with 3,415 more rows, and 7 more variables: diabetes <int>,
## #   diabp <dbl>, sysbp <dbl>, height <int>, weight <int>, smoke <int>,
## #   racebw <int>
```

CHD Data

- Change `gender` to `female`, remove leading column and `gender`.

```
dat <- dat %>%
  mutate(female = factor(gender)) %>%
  select(-gender, -X1)
```

CHD Data: Replace missing values with means for those covariates.

```
######  Check NAs
sum(is.na(dat))
```

```
## [1] 41
```

```
dat %>% filter(!complete.cases(.))
```

```
## # A tibble: 41 x 16
##       idno choltot incchd   hdl alcoh   bmi  trig   age diabetes diabp sysb
##      <int>   <dbl>  <int> <int> <dbl> <dbl> <int> <int>    <int> <dbl> <dbl
##  1 3.02e6    222.       0    57 NA     25.8   249    66        1    75     9
##  2 3.50e6    231.       0    42 NA     22.8   182    66        1    81    15
##  3 4.00e6    256.       0    65 NA     32.8   208    66        2    75    12
##  4 5.04e6    247.       0    63 NA     37.1   148    66        1    72    12
##  5 3.04e6    226.       0    53  0.25  NA      105    67        1    77    12
##  6 6.04e6    290.       1    42 NA     22.4   251    67        1    68    12
##  7 3.03e6    213.       0    51 NA     34.9   103    68        2    69    12
##  8 3.04e6    246.       0    47 NA     28.0   212    68        1    50    13
##  9 3.50e6    286.       0    73 NA     35.4   150    68        1    98    16
## 10 3.50e6    249.       0    29 NA     26.6   197    68        2    74    13
## # ... with 31 more rows, and 5 more variables: height <int>, weight <int>,
## #   smoke <int>, racebw <int>, female <fct>
```

Save cleaned CHD Data

```r
write_csv(dat, "dat/chddata_dsi_cleaned.csv")
dat <- read_csv("dat/chddata_dsi_cleaned.csv")
```

```
## Parsed with column specification:
## cols(
##   idno = col_integer(),
##   choltot = col_double(),
##   incchd = col_integer(),
##   hdl = col_integer(),
##   alcoh = col_double(),
##   bmi = col_double(),
##   trig = col_integer(),
##   age = col_integer(),
##   diabetes = col_integer(),
##   diabp = col_double(),
##   sysbp = col_double(),
##   height = col_integer(),
##   weight = col_integer(),
##   smoke = col_integer(),
##   racebw = col_integer(),
##   female = col_integer()
## )
```

Using `qwraps2` to Summarize Data

```r
library(qwraps2)
data_summary <-
  list("Total Chol." =
         list("min" = ~min(choltot),
              "max" = ~max(choltot),
              "mean (sd)" = ~qwraps2::mean_sd(choltot)),
       "Inc. CHD" = list("% (n)" = ~qwraps2::perc_n(incchd)),
       "HDL" =
         list("min" = ~min(hdl),
              "max" = ~max(hdl),
              "mean (sd)" = ~qwraps2::mean_sd(hdl)),
       "BMI" =
         list("min" = ~min(bmi),
              "max" = ~max(bmi),
              "mean (sd)" = ~qwraps2::mean_sd(bmi)))

orig_opt <- options()$qwraps2_markup
options(qwraps2_markup = "markdown")
summary_table(dat, data_summary)
```

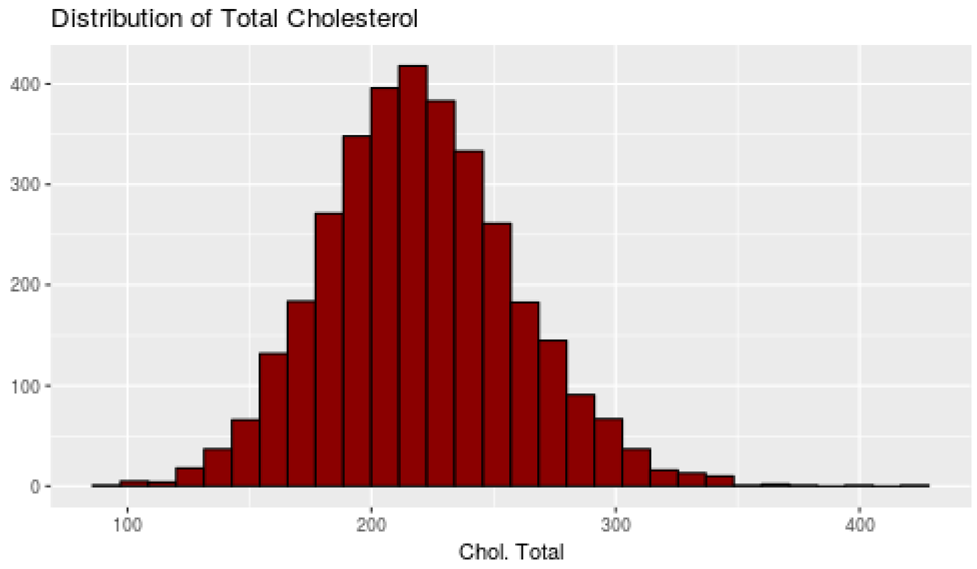
Using `qwraps2` to Summarize Data

	dat (N = 3425)
Total Chol.	
min	90.59
max	421.87
mean (sd)	219.88 ± 39.48
Inc. CHD	
% (n)	13.66% (n = 3,425)
HDL	
min	15
max	149
mean (sd)	54.90 ± 15.53
BMI	
min	14.65
max	49.41

Exploratory Plots

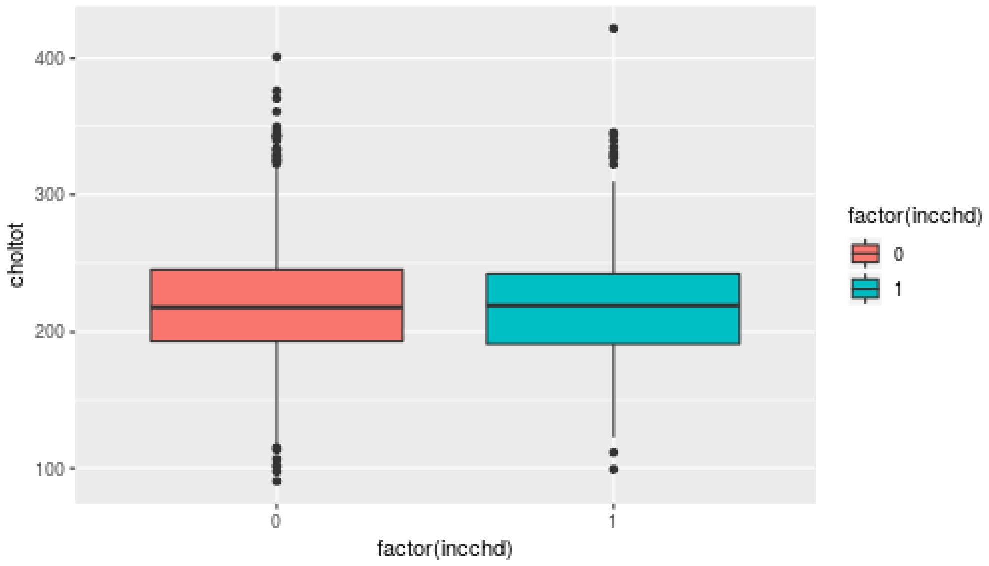
```
qplot(choltot, data = dat, fill = I("darkred"), color = I("black"), x
```

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



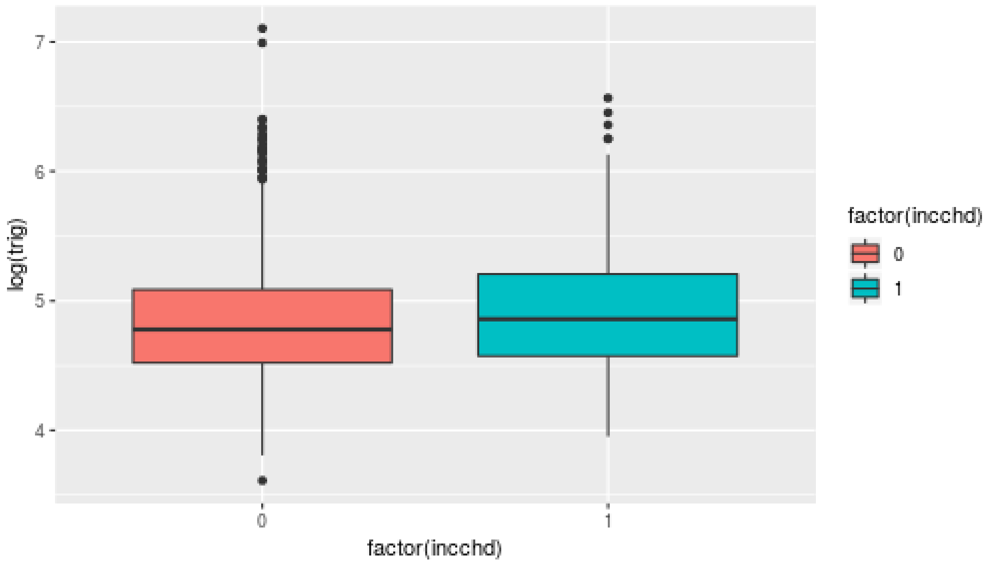
Exploratory Plots

```r
qplot(x = factor(incchd), y = choltot, data = dat, geom = "boxplot",
```



Exploratory Plots

```r
qplot(x = factor(incchd), y = log(trig), data = dat, geom = "boxplot'
```



Bootstrap Estimation

`purrr::map(x, f)` Applies a function f to each element of a list x

```r
a <- 1:10
purrr::map(a, function(x) x^2)
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] 4
##
## [[3]]
## [1] 9
##
## [[4]]
## [1] 16
##
## [[5]]
## [1] 25
##
## [[6]]
## [1] 36
```

Bootstrap Estimation

New Function: `purrr::map`

`purrr::map(x, f)` Applies a function f to each element of a list x

```r
a <- 1:10
purrr::map_dbl(a, function(x) x^2)
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```r
a <- 1:10
purrr::map_dbl(a, ~ .^2)
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

Bootstrap Estimation

```r
boot_mean <- function(d, i) {
  mean(d[i])
}
```

```r
dat_booted <- dat %>%
  dplyr::group_by(female) %>%
  tidyr::nest()
dat_booted
```

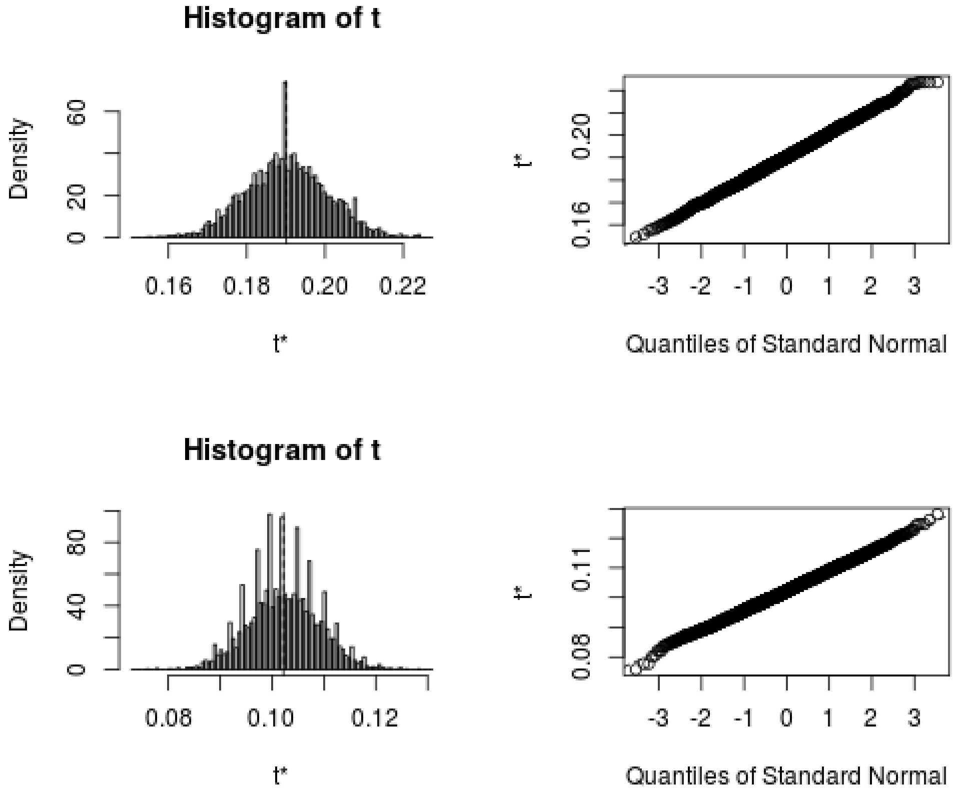
```
## # A tibble: 2 x 2
##   female data
##    <int> <list>
## 1      1 <tibble [1,341 × 15]>
## 2      0 <tibble [2,084 × 15]>
```

Bootstrap Estimation

```
dat_booted <- dat_booted %>%
  dplyr::mutate(booted = purrr::map(.x = data,
                                    ~ boot::boot(data = .x$incchd,
                                                 statistic = boot_mea
                                                 R = 5000,
                                                 stype = "i")))
dat_booted
```

```
## # A tibble: 2 x 3
##   female data                       booted
##    <int> <list>                     <list>
## 1      1 <tibble [1,341 × 15]> <S3: boot>
## 2      0 <tibble [2,084 × 15]> <S3: boot>
```

Bootstrap Estimation



Bootstrap Estimation

```
dat_booted <- dat_booted %>%
  dplyr::mutate(booted_ci = purrr::map(.x = booted,
                                       ~ boot::boot.ci(.x,
                                                       conf = 0.95,
                                                       type = "bca"))
dat_booted
```

```
## # A tibble: 2 x 4
##   female data                       booted     booted_ci
##    <int> <list>                     <list>     <list>
## 1      1 <tibble [1,341 × 15]> <S3: boot> <S3: bootci>
## 2      0 <tibble [2,084 × 15]> <S3: boot> <S3: bootci>
```

Bootstrap Estimation

```
dat_booted <- dat_booted %>%
    dplyr::mutate(statistic = purrr::map(.x = booted_ci,
                                         ~ .x$t0),
                  lower_ci = purrr::map(.x = booted_ci,
                                        ~ .x$bca[[4]]),
                  upper_ci = purrr::map(.x = booted_ci,
                                        ~ .x$bca[[5]])) %>%
  dplyr::select(-data, -booted, -booted_ci) %>%
  tidyr::unnest()
dat_booted
```

```
## # A tibble: 2 x 4
##   female statistic lower_ci upper_ci
##    <int>     <dbl>    <dbl>    <dbl>
## 1      1     0.190   0.170     0.210
## 2      0     0.102   0.0893    0.115
```

Bootstrap Estimation

```r
boot_mean <- function(d, i) {
  mean(d[i])
}

dat_booted <- dat %>%
  dplyr::group_by(female) %>%
  tidyr::nest() %>%
  dplyr::mutate(booted = purrr::map(.x = data,
                                    ~ boot::boot(data = .x$incchd,
                                                 statistic = boot_mea
                                                 R = 5000,
                                                 stype = "i"))) %>%
  dplyr::mutate(booted_ci = purrr::map(.x = booted,
                                       ~ boot::boot.ci(.x,
                                                       conf = 0.95,
                                                       type = "bca")
  dplyr::mutate(statistic = purrr::map(.x = booted_ci,
                                       ~ .x$t0),
                lower_ci = purrr::map(.x = booted_ci,
                                      ~ .x$bca[[4]]),
                upper_ci = purrr::map(.x = booted_ci,
                                      ~ .x$bca[[5]])) %>%
```

Boostrap Estimation

```
knitr::kable(x = dat_booted, digits = 3, col.names = c("Female", "Est
```

95% CI for CHD Incidence by Gener

Female	Est. CHD Incidence	Lower 2.5%	Upper 97.5%
1	0.190	0.169	0.211
0	0.102	0.089	0.116

Tidy Modeling with broom

```r
library(broom)
fit_glm <- glm(incchd ~ ., data = dat, family = "binomial")
tidy(fit_glm)
```

```
## # A tibble: 16 x 5
##    term         estimate    std.error statistic    p.value
##    <chr>           <dbl>        <dbl>     <dbl>      <dbl>
##  1 (Intercept) -1.19e+1 4.29             -2.77   0.00553
##  2 idno         4.80e-8 0.0000000456      1.05   0.293
##  3 choltot      3.90e-3 0.00142           2.75   0.00596
##  4 hdl         -6.29e-3 0.00461          -1.36   0.173
##  5 alcoh       -3.16e-2 0.0106           -2.97   0.00299
##  6 bmi          9.70e-2 0.0751            1.29   0.197
##  7 trig         6.76e-4 0.000712          0.950  0.342
##  8 age          3.20e-2 0.00986           3.25   0.00117
##  9 diabetes     2.59e-1 0.0687            3.76   0.000168
## 10 diabp       -6.56e-3 0.00549          -1.20   0.232
## 11 sysbp        1.18e-2 0.00284           4.15   0.0000328
## 12 height       2.87e-2 0.0251            1.14   0.252
## 13 weight      -1.61e-2 0.0125           -1.28   0.199
## 14 smoke        2.00e-1 0.0791            2.53   0.0115
## 15 racebw      -1.86e-2 0.257            -0.0725 0.942
```

Tidy Modeling with `broom`

```r
knitr::kable(tidy(fit_glm) %>%
               filter(p.value < 0.05, term != "(Intercept)") %>%
               select(-statistic) %>%
               transmute(Source = term, Estimate = estimate, SE = std
             format = "markdown",
             digits = 4)
```

Source	Estimate	SE	P
choltot	0.0039	0.0014	0.0060
alcoh	-0.0316	0.0106	0.0030
age	0.0320	0.0099	0.0012
diabetes	0.2587	0.0687	0.0002
sysbp	0.0118	0.0028	0.0000
smoke	0.1999	0.0791	0.0115
female	0.7720	0.1655	0.0000

Intro to R Markdown

Intro to R Markdown

- **R Markdown** is an implementation of the *Markdown* document formatting language
- Markdown is a versatile tool that makes it easy to make readable scientific documents in a variety of formats
- R markdown is actively developed and supported by the RStudio team, which means:
    - RStudio has many tools and features to make R Markdown flexible and easy to use
    - New R Markdown features and packages are frequently released

Intro to R Markdown

- **R For Data Science** on the intent of R Markdown:

R Markdown files are designed to be used in three ways:

1. *For communicating to decision makers, who want to focus on the conclusions, not the code behind the analysis.*

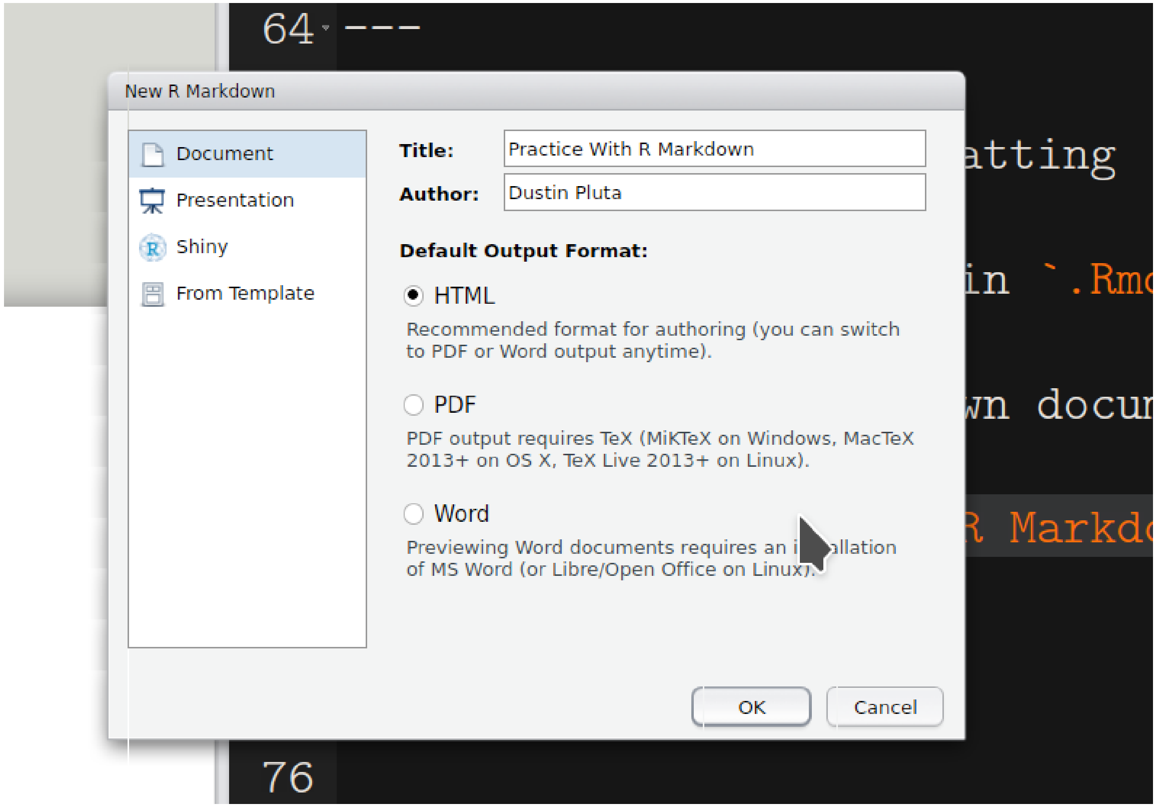
2. *For collaborating with other data scientists (including future you!), who are interested in both your conclusions, and how you reached them (i.e. the code).*

3. *As an environment in which to do data science, as a modern day lab notebook where you can capture not only what you did, but also what you were thinking.*

Getting Started with R Markdown

- R Markdown files end in `.Rmd`
- Create a new R markdown document in RStudio:
    - `File > New File > R Markdown...`

Getting Started with R Markdown



New R Markdown

- Document
- Presentation
- Shiny
- From Template

Title: Practice With R Markdown

Author: Dustin Pluta

Default Output Format:

[✓ HTML]
Recommended format for authoring (you can switch to PDF or Word output anytime).

PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Getting Started with R Markdown

- The default R Markdown template gives some examples of basic R Markdown features

Getting Started with R Markdown

Getting Started with R Markdown

Getting Started with R Markdown

Getting Started with R Markdown

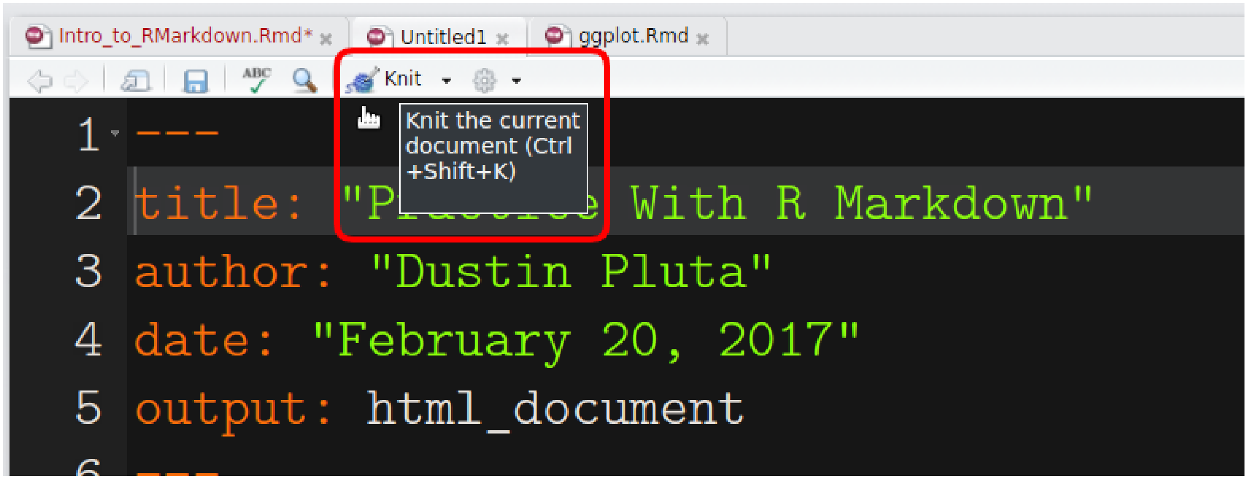
Getting Started with R Markdown

Getting Started with R Markdown

Getting Started with R Markdown

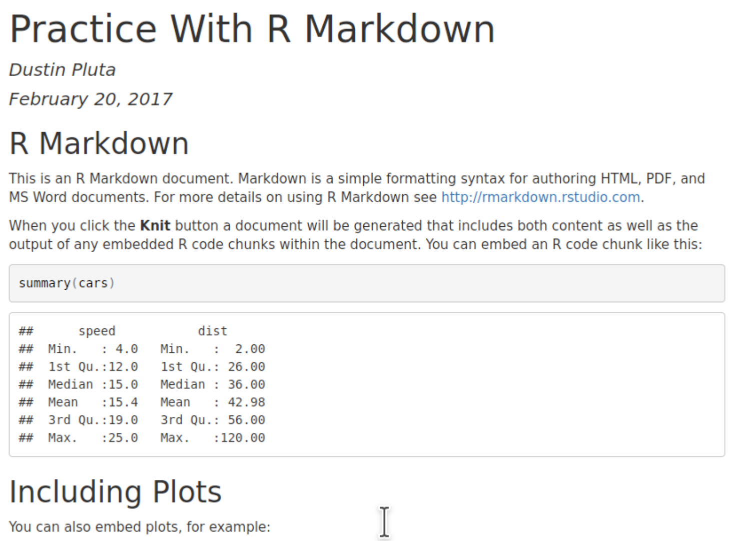
Getting Started with R Markdown

- Compile or "knit" the R Markdown document to the desired format (either html, pdf, or Word document)



```
---
title: "Pr      e With R Markdown"
author: "Dustin Pluta"
date: "February 20, 2017"
output: html_document
```

Getting Started with R Markdown



Practice With R Markdown

Dustin Pluta

February 20, 2017

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

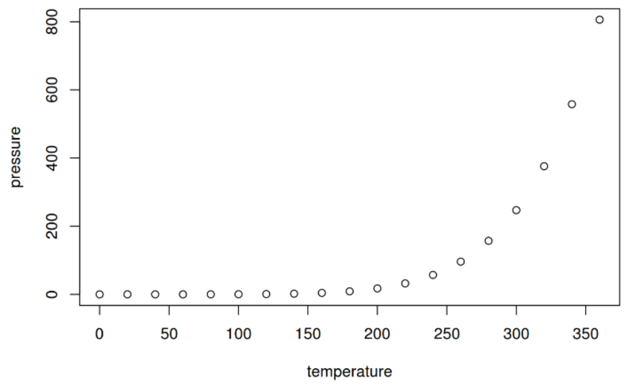
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00  
```

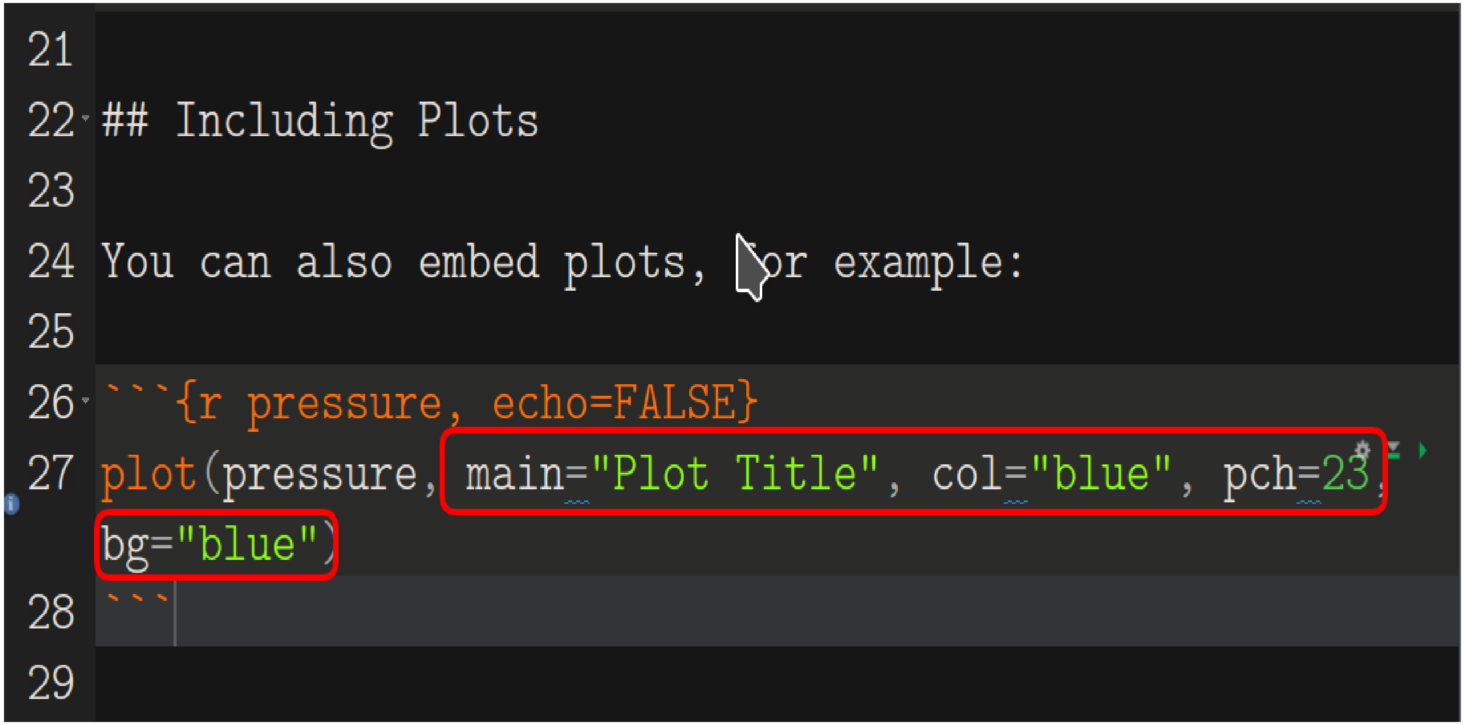
Including Plots

You can also embed plots, for example:



Getting Started with R Markdown

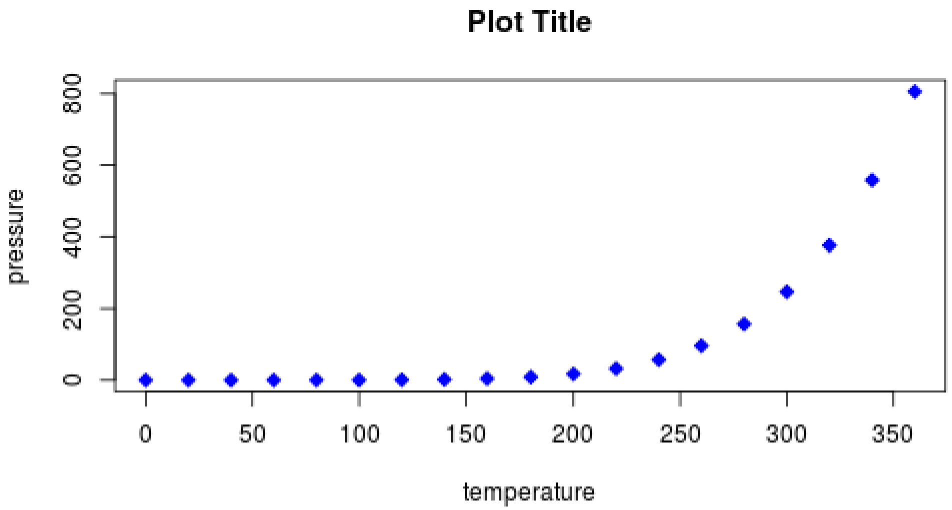
- Let's modify the plot to include a title, and make the points blue.



~~~
## Including Plots

You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure, main="Plot Title", col="blue", pch=23,
bg="blue")
```

~~~

Getting Started with R Markdown

```
plot(pressure, main = "Plot Title", pch = 23, col = "blue", bg = "blu
```



Practice With R Markdown

R Markdown Cheat Sheet

1. R Markdown Basic Example: Shows simple plot with `cars` data set.

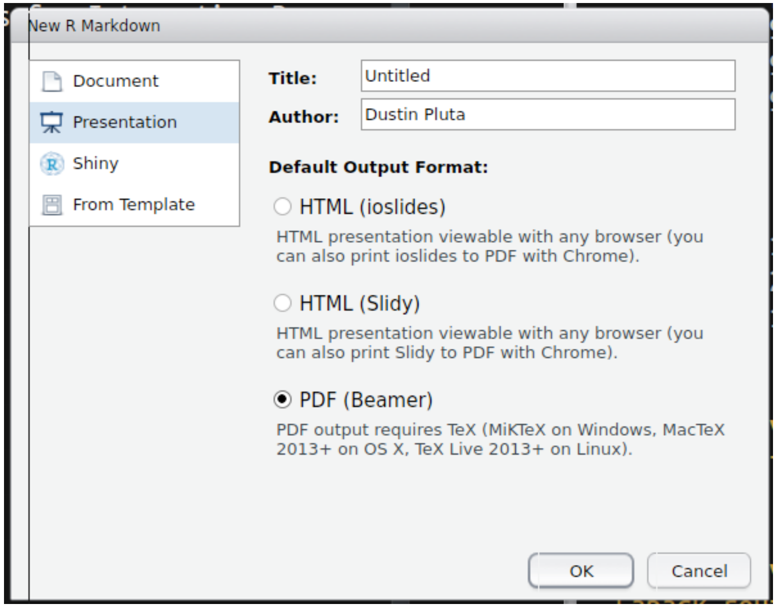
2. R Markdown Exercise Set 1: More examples and some exercises to try on the Iris and IMDB data sets.

More R Markdown Features

1. Presentations: beamer, ioslides, slidy, xaringan
2. knitr
3. Blogdown
4. Bookdown
5. Interactive Documents

Presentations

- You can easily create academic presentations using 4 different formats
  - beamer (pdf)
  - ioslides (html)
  - slidy (html)
  - xaringan (html)


New R Markdown

Title: Untitled
Author: Dustin Pluta

Default Output Format:

( ) HTML (ioslides)
HTML presentation viewable with any browser (you can also print ioslides to PDF with Chrome).

( ) HTML (Slidy)
HTML presentation viewable with any browser (you can also print Slidy to PDF with Chrome).

[✓ PDF (Beamer)]
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

knitr

- R Markdown can make full use of Latex through the `knitr` package
- `knitr` lets you easily display mathematical formulas and other Latex formatting in your Markdown document
- For example, math can be inserted inline like $\alpha^2 + \beta^2 = \gamma^2$ or in display mode:

$$Y = X\beta + \varepsilon$$

$$\int_{\mathbb{R}} \sum_{i=1}^{n} \nabla \ell_i d\mu$$

Blogdown

Making a Website Using Blogdown, Hugo, and GitHub pages/

Example Blogdown Blog: Simply Statistics

Next Steps

- `RMarkdown_Basic_Example.Rmd`
- `Iris_Example.Rmd`
- `RMarkdown_Exercise_Set1.Rmd`
- `CHD_Data_Analysis.Rmd`

Some Resources for R

- `dplyr` and Data Wrangling Cheat Sheet
- R Markdown Cheat Sheet
- Data Carpentry Lessons for R
- `dplyr` Tutorial
- Advanced R
- R for Data Science
- Coursera Data Science Specialization