Introduction to ggplot

Andrew Bell

Learning Objectives

- Review Rzero
- Understand ggplot syntax
- Learn the grammar of graphics
- Start adding to your data visualization toolset

Review RZero

In essence, most everything you do in R involves **objects**. **Functions** take **arguments** (some of which are objects) and produce outputs (which you can assign to a new object).

Data frames are made up of multiple lists. You can create dataframes using the data_frame() function.

```
#new object "my_kids" assigned the output of the function "c". The function c creates a list of strings

my_family <- c("Katie","Finn","Jack","Josie")
kid_ages <- c(NA,1,3,5)
df <- data_frame(my_family, kid_ages)
```

```
## Warning: ‘data_frame()‘ was deprecated in tibble 1.1.0.
## Please use ‘tibble()‘ instead.
```

```
df
```

```
## # A tibble: 4 x 2
##   my_family kid_ages
##   <chr>        <dbl>
## 1 Katie           NA
## 2 Finn             1
## 3 Jack             3
## 4 Josie            5
```

- **Operators** - can manipulate **objects** (this manipulation can be assigned to a **new object**). To reference specific columns within a dataframe use the $ symbol (i.e. dataframe$columnname).

```
#new object "how_old_my_kids_will_be_in_5_years" is assigned to the output of kids_ages plus 5

how_old_my_kids_will_be_in_5_years <- df$kid_ages + 5

how_old_my_kids_will_be_in_5_years
```

```
## [1] NA  6  8 10
```

- **Functions** - accept **objects** and returns the result of the function (which can be assigned to a **new object**). NA values can be the bane of any R coder's existence. Try running the code below without the argument na.rm = TRUE, what does the function return? What do you think na.rm does?

```
#new object "average_age" is assigned to the output the function mean() which is accepting the kid_ages

average_age <- mean(kid_ages, na.rm = TRUE)

average_age
```

```
## [1] 3
```

At this point, if you don't feel comfortable with the differences between objects, functions, arguments, and operators go back to Rzero and work through that lesson again.

Learning ggplot syntax

We want to be able to make visualizations from data - so we need a function that outputs a plot based on the objects and arguments we give it. Enter the **ggplot()** function. Below is most basic syntax (structure of code required to make the function work) of the ggplot function.

`ggplot(data = <DATA>) +   <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))`

We are going to use the friends_info dataset we used at the end of Rzero to get started with ggplot().

First let's take a look at our friends_info. Note how many variables and observations are found in the dataset.

```
friends_info <- read_csv("Data/friends_info.csv")
```

```
## Rows: 236 Columns: 8

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): title, directed_by, written_by
## dbl  (4): season, episode, us_views_millions, imdb_rating
## date (1): air_date

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
friends_info
```

```
## # A tibble: 236 x 8
##    season episode title     directed_by  written_by  air_date   us_views_millio~
##     <dbl>   <dbl> <chr>     <chr>        <chr>       <date>                <dbl>
##  1      1       1 The Pilot James Burro~ David Cran~ 1994-09-22             21.5
##  2      1       2 The One ~ James Burro~ David Cran~ 1994-09-29             20.2
##  3      1       3 The One ~ James Burro~ Jeffrey As~ 1994-10-06             19.5
##  4      1       4 The One ~ James Burro~ Alexa Junge 1994-10-13             19.7
##  5      1       5 The One ~ Pamela Frym~ Jeff Green~ 1994-10-20             18.6
##  6      1       6 The One ~ Arlene Sanf~ Adam Chase~ 1994-10-27             18.2
##  7      1       7 The One ~ James Burro~ Jeffrey As~ 1994-11-03             23.5
##  8      1       8 The One ~ James Burro~ Marta Kauf~ 1994-11-10             21.1
##  9      1       9 The One ~ James Burro~ Jeff Green~ 1994-11-17             23.1
## 10      1      10 The One ~ Peter Bonerz Adam Chase~ 1994-12-15             19.9
## # ... with 226 more rows, and 1 more variable: imdb_rating <dbl>
```

```
head(friends_info)
```

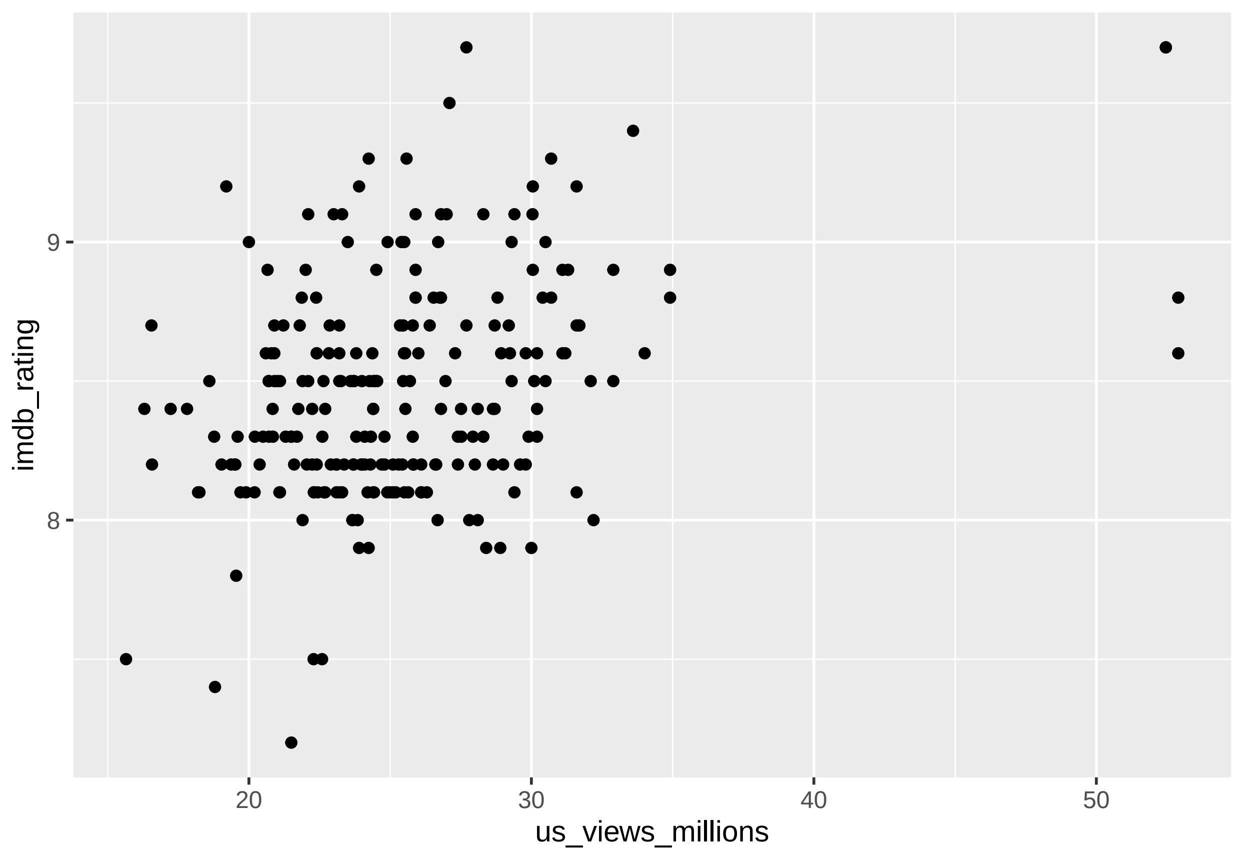
```
## # A tibble: 6 x 8
##   season episode title      directed_by written_by   air_date   us_views_millio~
##    <dbl>   <dbl> <chr>      <chr>       <chr>        <date>                <dbl>
## 1      1       1 The Pilot  James Burr~ David Crane~ 1994-09-22             21.5
## 2      1       2 The One w~ James Burr~ David Crane~ 1994-09-29             20.2
## 3      1       3 The One w~ James Burr~ Jeffrey Ast~ 1994-10-06             19.5
## 4      1       4 The One w~ James Burr~ Alexa Junge  1994-10-13             19.7
## 5      1       5 The One w~ Pamela Fry~ Jeff Greens~ 1994-10-20             18.6
## 6      1       6 The One w~ Arlene San~ Adam Chase ~ 1994-10-27             18.2
## # ... with 1 more variable: imdb_rating <dbl>
```

We want to create an object (first_viz) that contains a visualization created by the ggplot function. The ggplot function takes two essential arguments: data and the mapping of variables to specific aesthetics (like x and y coordinates). For our first visualization, we are going to plot the number of view (US_views_millions) by the critic ratings (imdb_rating)

```
first_viz <- ggplot(data = friends_info,
                    mapping = aes(x = us_views_millions,
                                  y = imdb_rating))
```

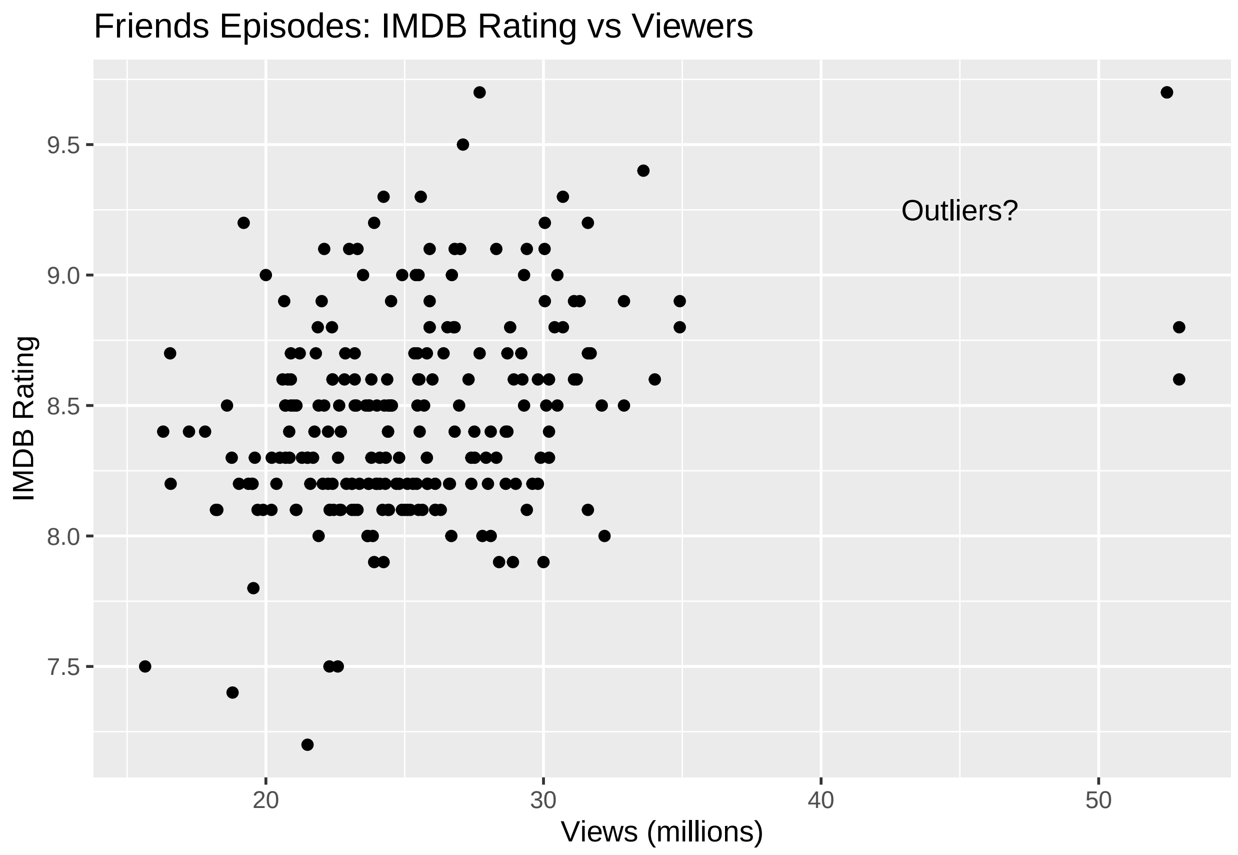
Our object now needs a geom layer (think plot type, e.g. scatterplot or bar plot) so we'll add that layer to our existing first_viz object.

```
first_viz <- first_viz + geom_point()

first_viz
```



We can continue to add layers to our visualization to do things like format the axis labels, change the axis scales, annotate specific elements of the plot area, etc.

```
first_viz <- first_viz +
  labs(title = "Friends Episodes: IMDB Rating vs Viewers",
       x = "Views (millions)",
       y = "IMDB Rating")+
  scale_y_continuous(breaks = seq(7,10.5, by = 0.5))+
  annotate("text", x = 45, y = 9.25, label = "Outliers?")

first_viz
```



Exercise: Creating a visualization

I want you to now create your own visualization that explores the how the shows popularity changed over time.

Instructions:

- 1 Create a new object that contains your visualization (use the friends_info dataset)
- 2 Use a geom to create a scatterplot that maps time to the x axis and USA views to the y axis
- 3 Make sure your visualization has a proper title and axis labels

```
#your code here
```

Learning all of ggplot's layers

This simply takes lots of practice. The discoverability of these layers requires knowledge of the options. One the major downsides to R but there are many resources to help.

Resources to help:

- ggplot cheatsheat
- Esquisser package as a GUI introduction to the ggplot syntax.

Esquisse This is a great tool to get students comfortable with ggplot's layer / syntax. Workshop to come…

One more thing… An Introduction to Data Transformation

What if we wanted to learn more about those those outliers that we identified above? We can use some basic data transformation functions to quickly filter our data to learn more abou those specific episodes.

```
filter_friends <- filter(friends_info, us_views_millions >40)

filter_friends
```

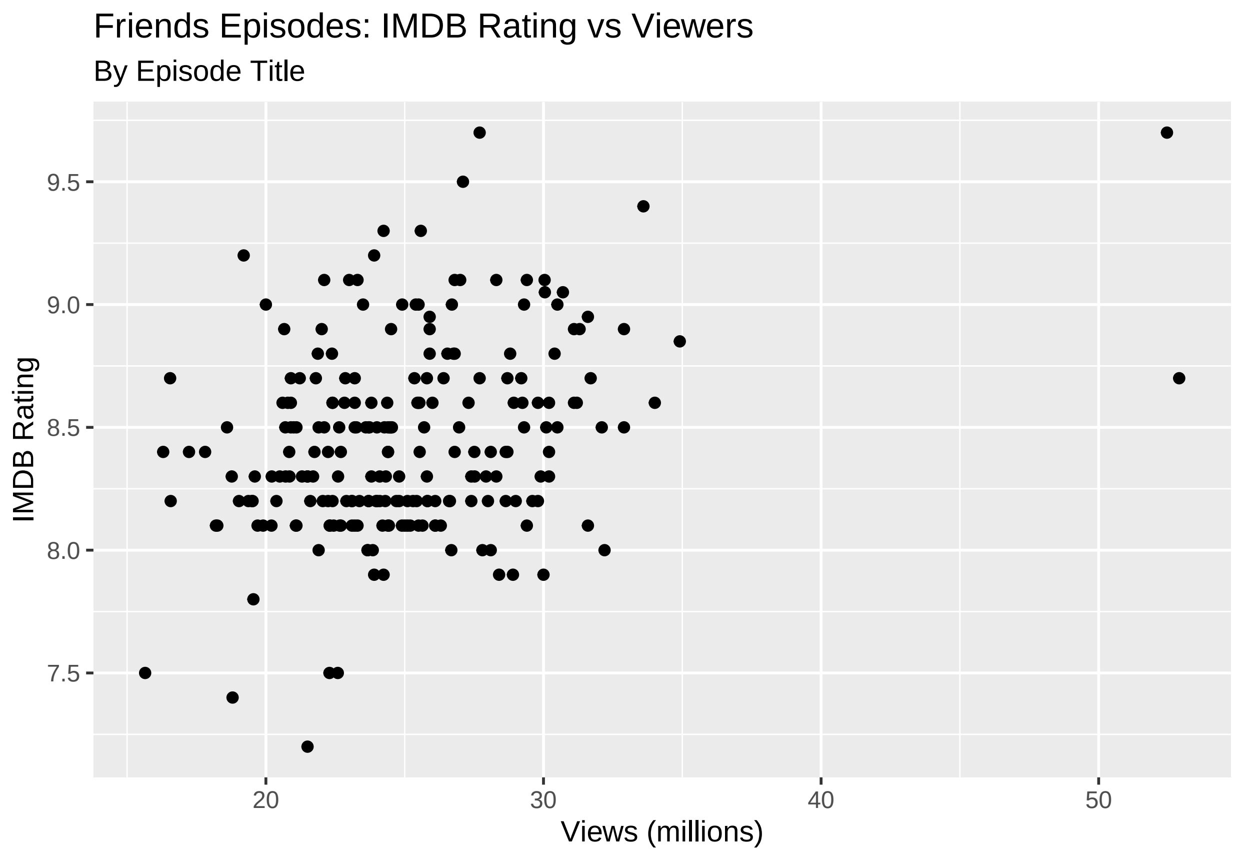
```
## # A tibble: 4 x 8
##   season episode title    directed_by  written_by    air_date   us_views_millio~
##    <dbl>   <dbl> <chr>    <chr>        <chr>         <date>                <dbl>
## 1      2      12 The One~ Michael Lem~ Jeffrey Astr~ 1996-01-28             52.9
## 2      2      13 The One~ Michael Lem~ Jeffrey Astr~ 1996-01-28             52.9
## 3     10      17 The Las~ Kevin S. Br~ Marta Kauffm~ 2004-05-06             52.5
## 4     10      18 The Las~ Kevin S. Br~ Marta Kauffm~ 2004-05-06             52.5
## # ... with 1 more variable: imdb_rating <dbl>
```

This brings up another issue: some 'titles' are made up of two episodes and when points overlap on our visualization, we can't see them. How could we fix this?

```
title_friends <- friends_info %>%
  group_by(title)%>%
  summarise(mean_views = mean(us_views_millions),
            mean_rating = mean(imdb_rating))

title_friends
```

```
## # A tibble: 227 x 3
##    title                              mean_views mean_rating
##    <chr>                                   <dbl>       <dbl>
##  1 The Last One                             52.5         9.7
##  2 The One After 'I Do'                     31.7         8.7
##  3 The One After Joey and Rachel Kiss       24.5         8.5
##  4 The One After Ross Says Rachel           31.1         8.9
##  5 The One After the Superbowl              52.9         8.7
##  6 The One After Vegas                      27.7         8.7
##  7 The One at the Beach                     28.8         8.8
##  8 The One Hundredth                        26.8         8.8
##  9 The One in Barbados                      25.5         8.6
## 10 The One in Massapequa                    22.0         8.2
## # ... with 217 more rows
```

```
second_viz <- ggplot(data = title_friends, mapping = aes(x = mean_views, y = mean_rating))+
  geom_point()+
  #geom_jitter()+
  labs(title = "Friends Episodes: IMDB Rating vs Viewers",
       subtitle = "By Episode Title",
       x = "Views (millions)",
       y = "IMDB Rating")+
  scale_y_continuous(breaks = seq(7,10.5, by = 0.5))

second_viz
```

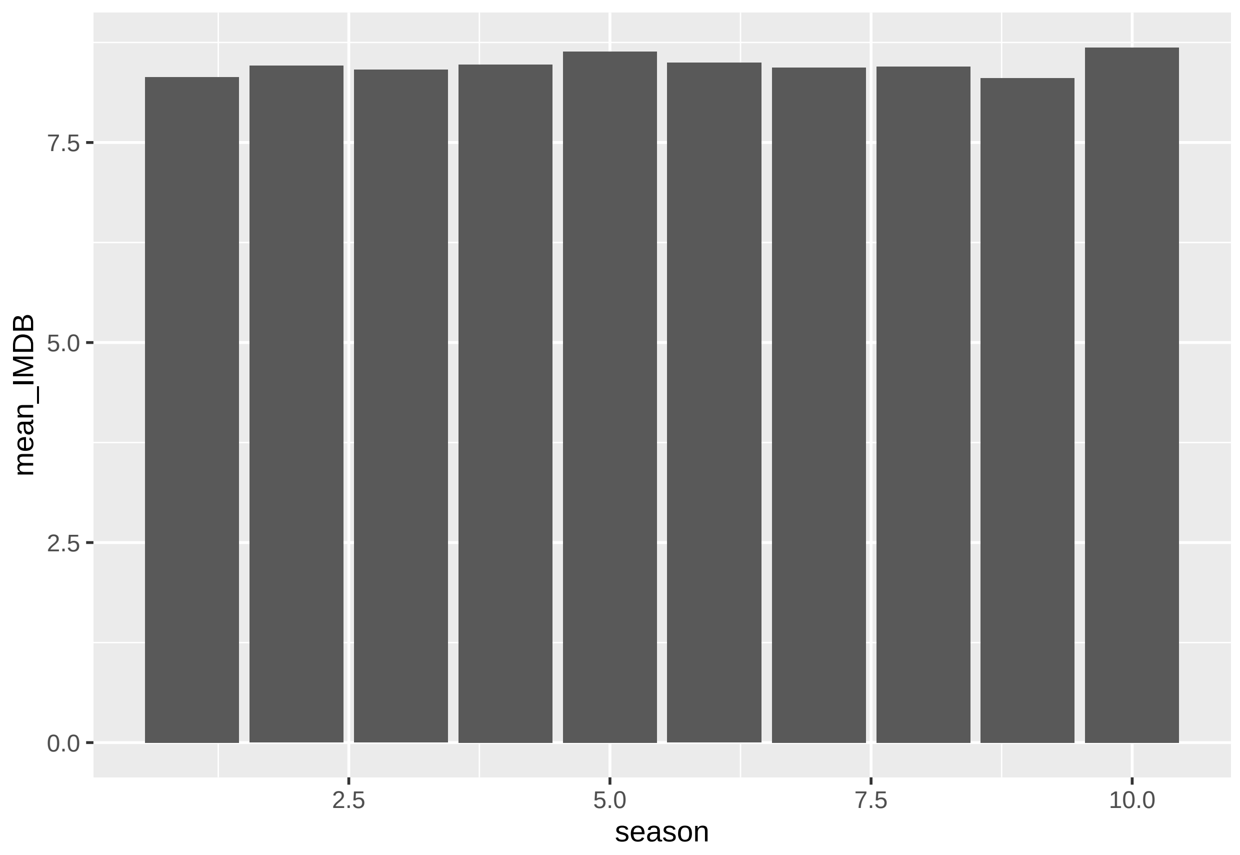


Suppose we want to look at the mean IMDB rating by season. What kind of data transformations and/or geom_layer might we need to run on the data?

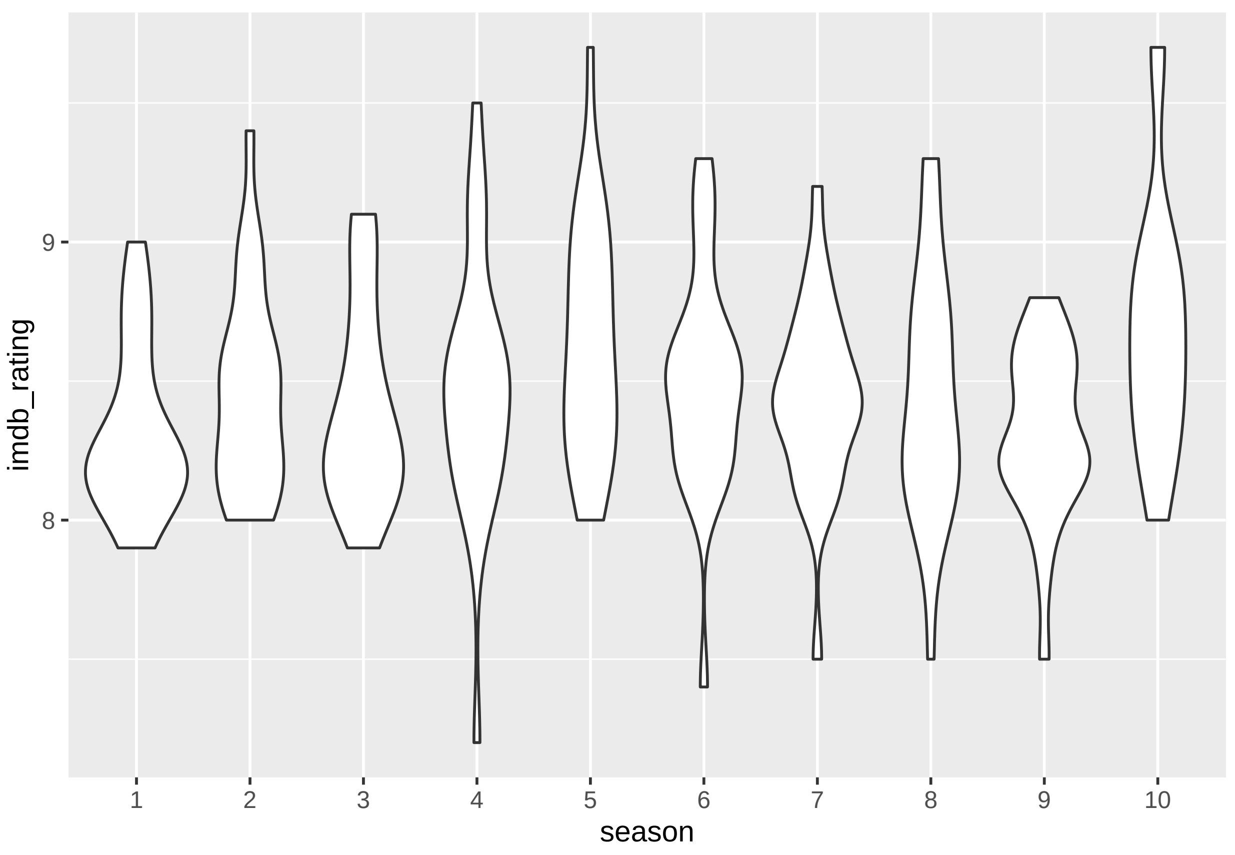
```
#option number 1 - generate the mean rating using dplyr functions and plot resulting values in bar plot
IMDB_byseason <- friends_info%>%
  group_by(season)%>%
  summarise(mean_IMDB = mean(imdb_rating))

IMDB_byseason
```

```
## # A tibble: 10 x 2
##    season mean_IMDB
##     <dbl>     <dbl>
##  1      1      8.32
##  2      2      8.46
##  3      3      8.41
##  4      4      8.48
##  5      5      8.64
##  6      6      8.50
##  7      7      8.44
##  8      8      8.45
##  9      9      8.30
## 10     10      8.69
```

```
ggplot(IMDB_byseason, aes(season, mean_IMDB))+
  geom_bar(stat = "identity")
```



```
#option number 2 - generate a boxplot or violin plot to look at the distribution of the ratings by seas

friends_info$season <- as.factor(friends_info$season)
ggplot(friends_info, aes(season, imdb_rating))+
  #geom_boxplot()+
  geom_violin()
```



9
8
imdb_rating
1
2
3
4
5
6
7
8
9
10
season