CS100: Studio 3

Introduction to Visualization in R

September 25 and 26, 2019

Instructions

During today’s studio, you will be creating data visualizations in R. Please write all of your code, and answers to the questions, in an R markdown document.

Upon completion of all tasks, a TA will give you credit for today’s studio. If you do not manage to complete all the assigned work during the studio period, do not worry. You can continue to work on this assignment until the following Wednesday at 6 PM. Come by TA hours any time before then (or, for this week only, at the start of next week’s studio, because some TA hours were cancelled) to show us your completed work and get credit for today’s studio.

Objectives

By the end of this studio, you will know:
- How to use R base graphs to visualize data
- How to use ggplot to visualize data

Part 1: Visualizing Movie Ratings

FiveThirtyEight published the article Be Suspicious Of Online Movie Ratings, Especially Fandango’s, where it was reported that, for the same movies, Fandango had consistently higher ratings than other sites, such as IMDb, Rotten Tomatoes, and MetaCritic. FiveThirtyEight also reported that Fandango was inflating users’ true ratings by rounding them up to the nearest half star (e.g., 4.1 stars would be rounded up to 4.5 stars). Fandango may have been motivated to implement this rounding scheme because it not only provides movie ratings, but also sells movie tickets. When people see higher ratings for a movie, they might be more inclined to go see it.

Data

FiveThirtyEight publicly released the data they used in their analysis, which you can view and download here. Additionally, documentation can be found here; please visit this web page to see the various variables and their definitions. In this studio, you will replicate FiveThirtyEight’s visualizations and findings using R.

Setup

To complete this studio, you will need to install a new R library called GGally, which extends the functionality of ggplot. You’ll be using it today, along with ggplot, to visualize the movie ratings data.
Open RStudio, and then run the following command in the console:

```r
install.packages("GGally")
```

Insert the following code chunk at the start of an R markdown file, and then run it by clicking ‘Run’ and selecting ‘Run All’. Make sure to include the three apostrophes before and after the code, as this tells R where the code chunk begins and ends.

```r
#r setup, include = FALSE
library(dplyr)
library(ggplot2)
library(GGally)

movie_scores <- read.csv("http://cs.brown.edu/courses/cs100/studios/data/3/Fandango_wide.csv")
```

Getting started

The ggplot syntax for a basic, geometric plot, like a bar graph or simply points on the Cartesian plane, is as follows:

```r
ggplot(data = your data here, aes(x = x, y = y)) + geom_X()
```

Here data is a data frame, aes is the aesthetic mappings (where the x and y values are defined, and other properties, like color and fill, can be set), and geom_X is a geometric object, such as geom_bar or geom_point.

Beyond this basic syntax, you can build layers upon layers in a ggplot to add titles, legends, etc. Each additional layer is added using the + sign.

You will get lots of practice using ggplot in today’s studio.

Histograms

Let’s begin by investigating Fandango’s rating inflation.

Next, plot the distribution of movies’ ratings (i.e., number of stars out of 5). You can do so with base graphics, and with ggplot:

```r
hist(movie_scores$Fandango_Stars)
ggplot(data = movie_scores, aes(x = Fandango_Stars)) + geom_histogram()
```

Perhaps surprisingly, no movie on Fandango has fewer than 3 stars.

When you created your plots, you may have noticed that the following warning:

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

To match the intervals of Fandango’s star ratings, you should add an additional parameter to geom_histogram() that sets the binwidth to 0.5:

```r
geom_histogram(binwidth = 0.5)
```
Alternatively (and equivalently in this example), you can use `breaks` to specify precise bin boundaries, as follows:

```r
geom_histogram(breaks = c(2.5, 3, 3.5, 4, 4.5, 5))
```

For clarity, add a layer with a plot title and informative axes labels, using the following syntax:

```r
+ labs(title = 'your title here', x = 'your x-axis label here', y = 'your y-axis label here')
```

Now, using ggplot, create a histogram for “Fandango_Ratingvalue”. Can you see how the distribution of ratings differs from the distribution of stars. Perhaps not. How can you improve the visualizations to show this difference more clearly? Discuss this question with your partner before continuing.

One obvious way to more clearly visualize the difference is to plot the histogram of differences directly. Go ahead and do this, by creating a histogram of the “Fandango_Difference” variable, which measures how much “Fandango_Ratingvalue” was “rounded up” to reach the corresponding “Fandango_Stars”. What do you notice about this histogram?

*Hint:* Be sure to play around the bin width until this plot is intelligible.

Another, arguably better (in this case), way to visualize this difference is to overlay the two histograms, by plotting one on top of the other. The best way to accomplish this is by using the `alpha` parameter, which specifies the degree of transparency of the data in a plot. An `alpha` of 1 is completely opaque (the only visible data are the data plotted last), while an `alpha` of 0 is completely transparent: i.e., invisible. Create another plot by layering the both Stars and Ratings histograms on top of a basic ggplot layer that does nothing but define the data: i.e., `ggplot(data = movie_scores).

*Hint:* Use the `fill` parameter to color the histograms: e.g., `aes(fill = "red")`.

**Box Plots**

Next, let’s create box plots to compare the distribution of ratings among the different movie sites.

In `base` graphics, the command to generate a boxplot is `boxplot(y ~ grp)`, where `y` is a vector of numeric variables that is grouped by the factors in the `grp` vector. In this application, `y` should be the movie ratings (normalized to be on a scale of 0 to 5), and `grp` should be the various movie sites, so that we can plot the data using a command similar to:

```r
\`
\`
\`
```

Unfortunately, the current organization of the data is not amenable to this command. Specifically, the data are organized in `wide` form, which makes them
easy for people to interpret, but not so easy for computers to process. Recall the attendance database presented during lecture. There, the teacher took attendance in a table in wide form, with student names as rows and days of the week as columns. But, as mentioned in lecture, the days of the week are technically values, not variables; that is, “DayOfTheWeek” is the variable, with values Monday, Tuesday, etc. When data are organized with variables (only; no values) as columns, and observations as rows, they are said to be in long form. Fortunately, there are R libraries that automatically convert databases from wide to long form. You will learn to use one of these libraries, tidyr, in a few weeks when we turn to data cleaning. For now, we have done the conversion for you.

The long form of the Fandango data set is available here. Load the data into R in long form as follows:

```
```r
norm_ratings_long <- read.csv("http://cs.brown.edu/courses/cs100/studios/data/3/fandango_long.csv")
```

We’ve selected the columns of interest in the file, in this case “FILM”, “site”, and “rating”. Look at the data frame using the following command.

```
```r
View(norm_ratings_long)
```

As “site” is a variable in `norm_ratings_long`, there are now multiple observations per movie. We can now create the desired boxplots, using “rating” as our y and “site” as our grp, in both base graphics and ggplot:

```
boxplot(data = norm_ratings_long, rating ~ site)
``` ggplot(norm_ratings_long) + geom_boxplot(aes(x = site, y = rating))

The labels on the x axis are difficult, if not impossible, to read. One solution to this problem is to swap the axes, so that the box plots are horizontal instead of vertical. You can do this by appending `+ coord_flip()` to the end of the call to ggplot to create the box plot.

Alternatively, you can change the names of each of the labels (“Fandango_Ratingvalue”, ..., “RT_user_norm”) by adding the following layer:

```
scale_x_discrete(labels = c("Fandango", "Fandango Stars", "IMDB", "Metacritic", "Rotten Tomatoes", "Rotten Tomatoes Users"))
```

Note that we use `scale_x_discrete` instead of `scale_y_discrete` even though the labels are on the y-axis. This is because in our call to `aes`, we set `x = site`, and only afterwards swapped the axes using `coord_flip()`. Additionally, we use `scale_x_discrete` instead of `scale_x_continuous` because these labels are categorical.
Inspect your box plot. Do Fandango’s ratings differ from the others? If so, how?

**Scatter Plots**

As FiveThirtyEight pointed out, the ratings on Fandango do indeed tend to be higher than those on other sites. Next, let’s investigate whether there is at least a correlation between Fandango’s ratings and those of the other sites. For example, are well-rated movies on Fandango also well-rated on Rotten Tomatoes?

Run the code below:

```
plot(movie_scores$Fandango_Ratingvalue, movie_scores$RT_norm)
```

```
ggplot(data = movie_scores) + geom_point(aes(x = Fandango_Ratingvalue, y = RT_norm))
```

You should see that there is *not* a strong correlation between the ratings on Fandango and RottenTomatoes. There is a movie that simultaneously scores 1 on RottenTomatoes and 4.5 on Fandango!

How do ratings on Fandango compare to Metacritic and IMDb? Create those scatterplots now.

It doesn’t seem like the ratings on Fandango align too well with the other websites. But maybe the ratings are not correlated across the other sites either.

Let’s use the `pairs` function (in base graphics) to create a scatterplot matrix for our movie data:

```
ratings <- movie_scores %>% select(Fandango_Stars:IMDB_norm)
pairs(ratings)
```

To create a scatterplot matrix using ggplot, we use the `ggpairs` function in GGally:

```
ggpairs(ratings)
```

How do the Fandango ratings compare to those of the other websites? How do the ratings on the other 3 sites (IMDb, RottenTomatoes, Metacritic) compare to one another?

**Area Plots**

Finally, we will do our best to replicate the area plot found in the FiveThirtyEight
An *area* plot is a line plot, with the area below the line filled in.

To get started, copy and paste the following code into your R markdown file:

```r
ggplot(data = movie_scores) +
  geom_area(aes(x = Fandango_Stars, color = "fandango", fill = "fandango"), stat = "bin", binwidth = 0.5, alpha = 0.5) +
  geom_area(aes(x = RT_norm, color = "rt", fill = "rt"), stat = "bin", binwidth = 0.5, alpha = 0.25) +
  scale_fill_manual(values = c(fandango = "orangered", rt = "grey"), name = "Website", labels = c(fandango = "Fandango", rt = "Rotten Tomatoes", metacritic = "Metacritic")) +
  scale_color_manual(values = c(fandango = "orangered", rt = "grey"), guide = FALSE)
```

Extend this code with additional `geom_area` layers to plot the other movie sites as well. Then add a title, and informative labels on the x and y axes. When you think your visualization is complete, show your work to a TA.

In this studio, you experimented with only a few of the plot types available in ggplot2. To see examples of other plot types (e.g., `abline`, `dotplot`, `density`, etc.), here is a comprehensive resource. And here is the official ggplot cheat sheet.

**Part 2: Exploratory Data Analysis on Spotify Data**

At the end of each year, Spotify compiles a playlist of songs streamed that were most often over the course of that year. In the first studio, you explored these data in Google Sheets. In the time remaining, you should continue those
explorations in R, using the data visualization tools you learned about today. 

The Spotify dataset can be found here. Additionally, documentation can be found here. The audio features for each song were extracted using the Spotify Web API and the spotipy Python library. Credit goes to Spotify for calculating the audio feature values, and Nadin Tamer for populating this data set on Kaggle.

As usual, you can load the data into R like this:
```
{r setup, include = FALSE}
song_info <- read.csv("http://cs.brown.edu/courses/cs100/studios/data/3/top2018.csv")
```

Since the data set is large, you can save time by using dplyr to select particular variables on which to focus your investigations, and by filtering by song or song features (e.g., artist, danceability, etc.).

Try to generate two different types of visualizations that compare metrics among the top hits. You can use either R base plot or ggplot. Do your best to ensure that your graphs are visually appealing, easy to comprehend, and legible. Are there any interesting correlations between variables? Are there any surprises?

**End of Studio**

When you are done please call over a TA to review your work, and check you off for this studio. If you do not finish within the 2-hour studio period, remember to come to TA office hours to get checked off.