ezplot: How to Easily Make *ggplot2 Graphics* for Data Analysis

Guangming Lang

ezplot: How to Easily Make ggplot2 Graphics for Data Analysis

Guangming Lang

This book is for sale at http://leanpub.com/ezplot

This version was published on 2018-10-08

Tweet This Book!

Please help Guangming Lang by spreading the word about this book on Twitter!

The suggested tweet for this book is:

If you make data visualizations, you need to check out ezplot.

The suggested hashtag for this book is #ezplot.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#ezplot

Also By Guangming Lang

Score Personal Loan Applicants using R

Contents

Preface

This book will teach you two things: how to make good statistical charts, and how to do it fast. The tool we'll use is a R package called ezplot, which I wrote to help me with my consulting[1] work. A picture is worth a thousand words, but it's not easy to make a high quality chart, especially when we want to do it quickly. Hadley's ggplot is a great tool, but you still need to know the numerous commands for customizing in order to get a publishable chart. In the beginning, I used ggplot and did a lot of code recycling from project to project. But every time I copy-n-pasted a chunk of code, I had to change the data frame or variable name to make the code work for the new situation. This worked fine for one or two charts, but became very tedious when I needed to make more than 10 plots, and I often had to make 50+ charts. Even worse, as my code reservoir piled up, it became painful to find the right piece of code for the kind of customizations I wanted to do. Plus, code recycling made my scripts bulky and hard to read. So one day I sat down and wrote ezplot to change it all. Under the hood, all ezplot functions use ggplot2 functions. My goal was not to invent a new plotting system, but to make it very easy to create final-versioned ggplot2 charts with a couple of lines of code, requiring the user zero or minimal effort of customization.

Ezplot has made me happier. A plot that used to takes me 30 minutes now takes me less than 1 minute. I now use ezplot for all my client projects. I truly love it, and I think you'll love it too.

In this book, I'll show you the nuances of ezplot with example code and comments. You'll watch your productivity soar. After working through this book, you will be able to make the following 10 most used statistical charts in less than 1/30th of the time you use now.

- histogram & density plot
- boxplot
- qqplot
- barplot
- horizontal barplot & lollipop plot
- likert plot (a.k.a, horizontal diverging barplot)
- scatterplot
- lineplot
- dumbbell plot
- heatmap

You will get most out of this book by typing and running the code given in the book. Do NOT just copy and paste. Type the code. This will help you become a better R programmer. If you run into typos or errors, please let me know at gmlang@cabaceo.com.

Good luck and Happy Learning!

[1] http://www.cabaceo.com

Set up

1. Install R[2] and Rstudio[3].
2. Install a set of development tools:
    - On Windows, download and install Rtools[4].
    - On Mac, install the Xcode command line tools[5].
    - On Linux, install the R development package, usually called **r-devel** or **r-base-dev**.
3. Install the following R packages.

```
install.packages("tidyverse")
install.packages("devtools")
devtools::install_github("gmlang/ezplot")
```

Note: throughout this book, if when displaying plots, you encounter an error like this, `Error in grid.Call(L_textBounds, as.graphicsAnnot(x$label), x$x, x$y, : polygon edge not found`, just run `dev.off()` and then `print(p)`.

[2] http://www.r-project.org
[3] http://www.rstudio.com/products/rstudio/download/
[4] http://cran.r-project.org/bin/windows/Rtools/
[5] https://developer.apple.com/downloads

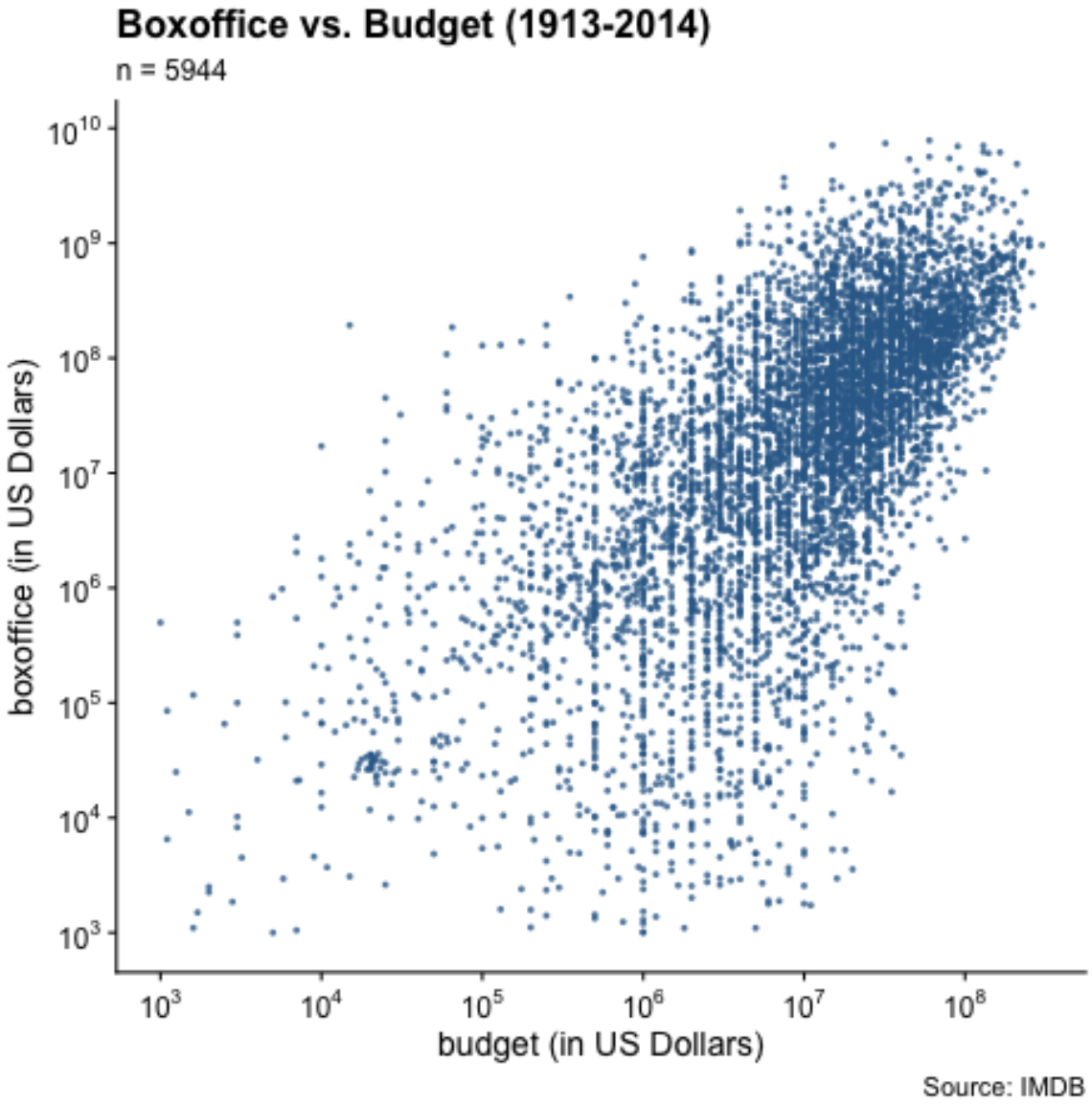
Scatter Plot

A scatter plot shows the relationship between two continuous variables. Let's apply the ezplot function `mk_scatterplot()` to the data frame `films` to get a function that we can use to make scatter plots for any two continuous variables in `films`.

```
library(dplyr)
library(ezplot)
plt = mk_scatterplot(films)
```

For example, we can use `plt()` to draw a scatter plot of `boxoffice` vs. `budget`. We'll use `log10` scale on both axes because the two variables are heavily right-skewed.

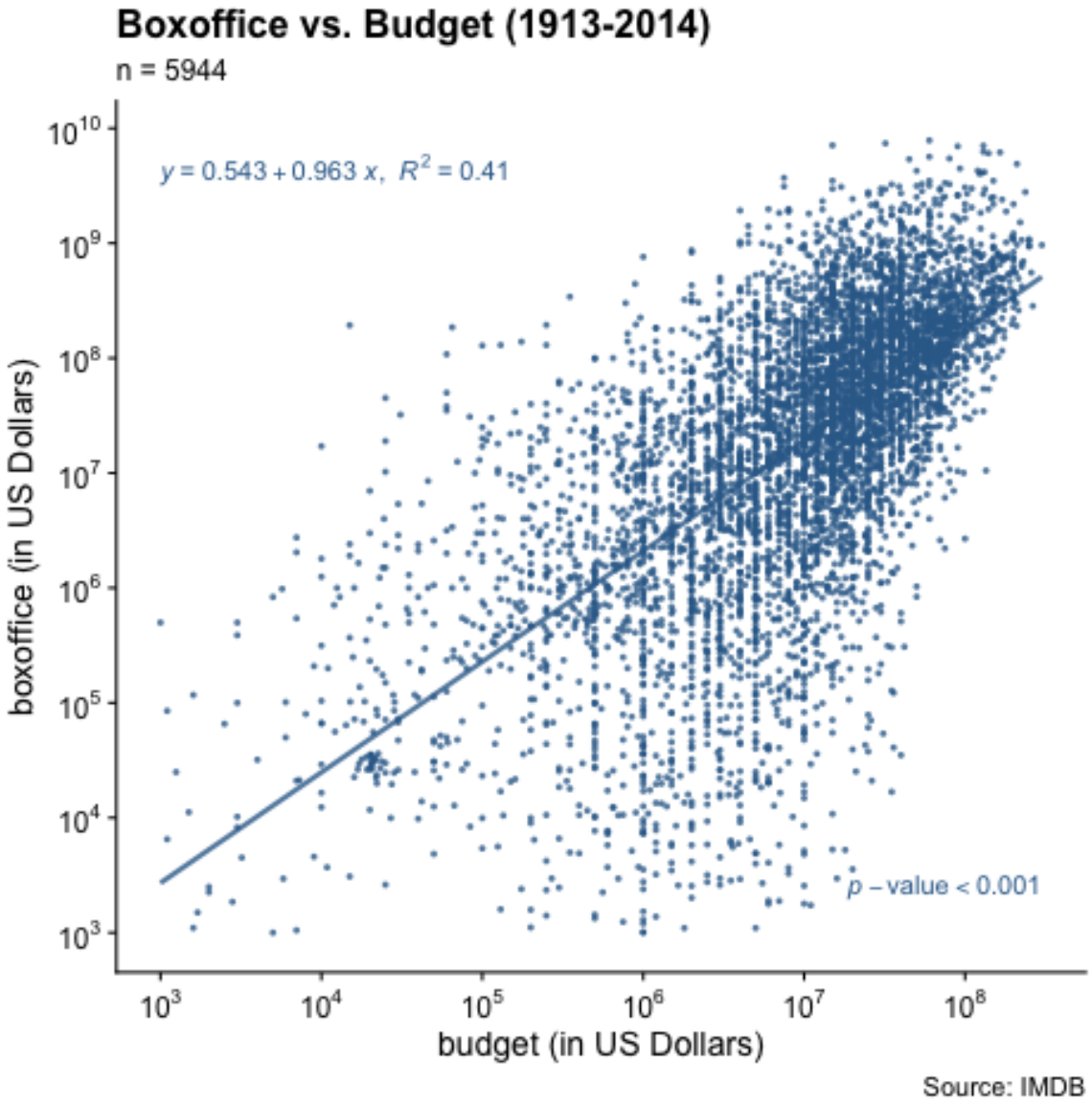
```
p = plt(xvar = "budget", yvar = "boxoffice") %>%
        add_labs(xlab="budget (in US Dollars)",
                 ylab="boxoffice (in US Dollars)",
                 title = "Boxoffice vs. Budget (1913-2014)",
                 caption = "Source: IMDB"
                 )
p = scale_axis(p, axis = "y", scale = "log10") # use log10 scale on y-axis
p = scale_axis(p, axis = "x", scale = "log10") # use log10 scale on x-axis
print(p)
```



Boxoffice vs. Budget

We see there's a clear positive linear trend between boxoffice and budget. What's the best line that summarizes this relationship? This is not an easy question. We need to run linear regression to find out. But luckily, we have the ezplot function `add_lm_line()`. It will add the best fitting line with its equation, R-squared and p-valued displayed on the plot. Let's add the best fitting line now.

```
add_lm_line(p)
```

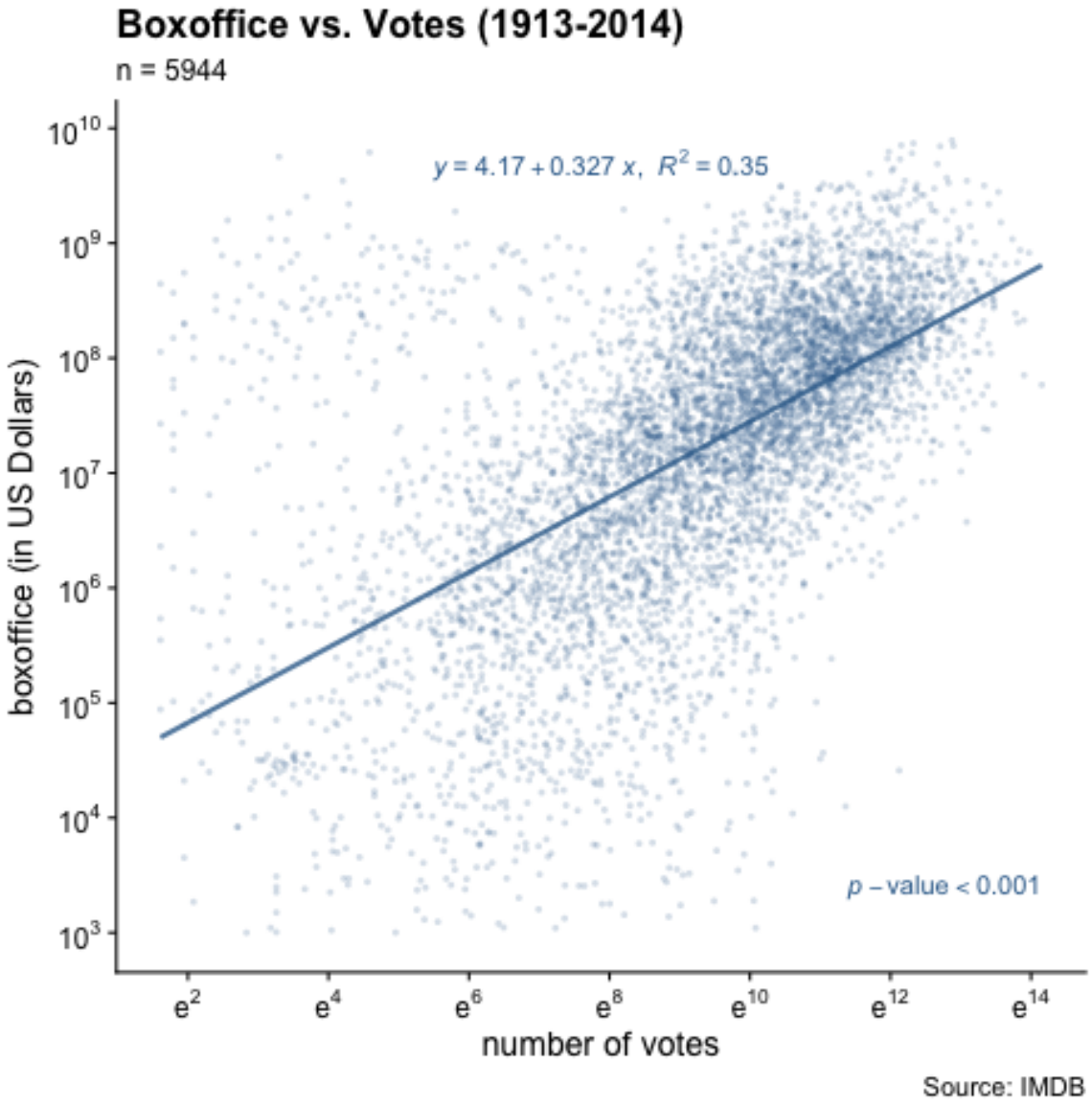


Boxoffice vs. Budget

The tiny p-value implies the linear relationship is statistically significant. The R-squared value implies 41% of the variation in boxoffice can be explained by the variation in budget (both at log10 scale).

The function `plt()` can be re-used. For example, we can use it to draw a scatter plot of `boxoffice` vs. `votes`.

```
p = plt("votes", "boxoffice", alpha = 0.2, jitter = T) %>%
        add_labs(xlab = "number of votes",
                 ylab = "boxoffice (in US Dollars)",
                 title = "Boxoffice vs. Votes (1913-2014)",
                 caption = "Source: IMDB"
                 )
p = scale_axis(p, "y", scale = "log10") # use log10 scale on y-axis
p = scale_axis(p, "x", scale = "log") # use log scale on x-axis
add_lm_line(p, eq_ypos = 0.95, eq_xpos = 0.5)
```

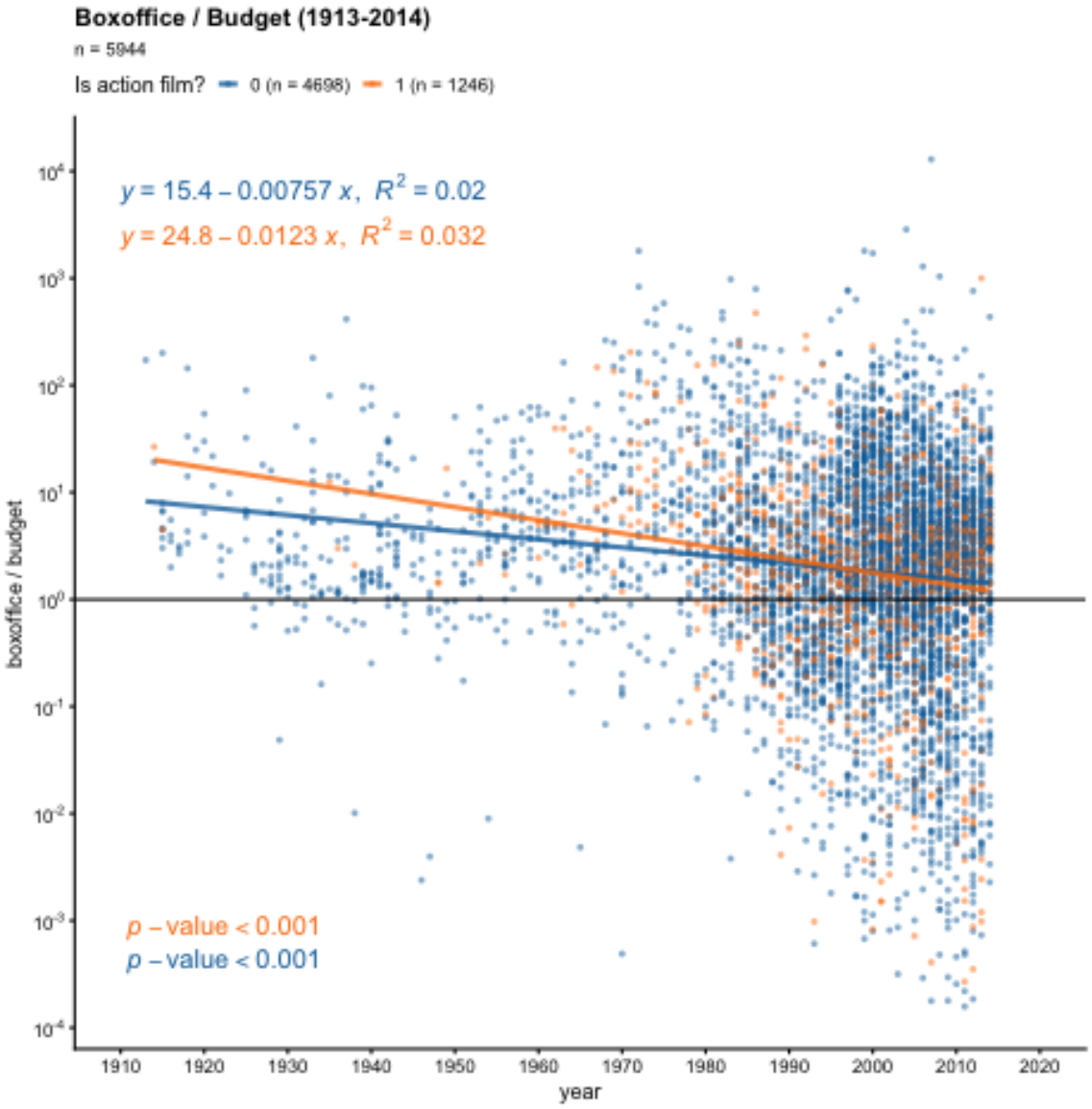


Boxoffice vs. Votes

We see there's also a strong linear relationship between boxoffice and votes. The tiny p-value implies this relationship is statistically significant.

We can also supply a categorical variable to color the data points. Consider this question: did action movies make money year after year? To answer it, we'll need to draw a scatter plot of `bo_bt_ratio` vs. `year` and color the points by the flagging variable `action`.

```
p = plt("year", "bo_bt_ratio", fillby = "action",
        legend_title = "Is action film?", legend_pos = "top",
        alpha = 0.5, font_size = 9) %>%
        add_labs(ylab = "boxoffice / budget",
                 title = "Boxoffice / Budget (1913-2014)")
p = p + ggplot2::geom_hline(yintercept = 1)
p = scale_axis(p, scale="log10")
add_lm_line(p, pval_xpos = "left")
```



Boxoffice vs. Budget colored by action film indicator

The orange dots are action films, while the blue dots are non-action films. First, notice there are more blue dots than orange dots. Second, the orange line has a steeper negative slope. If we pay attention to the orange dots before 1960, we'll see all orange dots before 1960 are above the y = 1 line, meaning action films always made money before 1960. But non-action films weren't as lucky. After 1960, some action films started losing money too.

Now it's your turn. Make scatter plots to answer the following questions:

1. Does drama make money year after year? What about comedy?
2. Is it true the higher the `rating`, the bigger the boxoffice/budget ratio (`bo_bt_ratio`)? What

about when viewed separartely under romance vs. non-romance films?

3. Is it true the more `votes` a film gets, the bigger its boxoffice/budget ratio? What about when viewed separartely under drama vs. non-drama films?