Intro Regression

Maria Tackett

Latest update: 2020-08-23

Contents

Welcome to Intro Regression!

The content in this book was originally developed for STA 210: Regression Analysis at Duke University.The computing aspects of the assignments are written using the `tidyverse` syntax in R; however, the assignments can be adapted to fit the computing language of your choice. All of the files are available in the Intro Regression GitHub repo.

This book is under development and will be periodically upated with new material. Please email me (maria.tackett@duke.edu) if you have any questions, feedback, or suggestions. I would also love to hear about your experience if you use any of the content in your course.

License

Chapter 1

Getting Started

1.1 How to use this book

Each chapter of this book is a topic that may be covered in an intermediate-level regression analysis course. The topics are arranged based on the way they were taught in STA 210: Regression Analysis; however, the assignments do not have to be used in the order they are presented. Feel free to use the text and adapt it to fit the needs of your course.

Each chapter includes several sections of assignments and supplmental notes about the mathematical details. Each section begins with one of the codes below to help you determine the type of assignment or note in that section:

- **COMP**: These assignments focus on the computing skills needed to conduct regression analysis. They were originally designed to be completed in groups in a weekly lab/discussion session; however, they can be also be used for homework assignments or in-class work days. Because the emphasis is computing, they include a lot of step-by-step instruction.

- **IN-CLASS**: Assignments to be completed as short in-class activities. Most of the code is already written, so students mostly run the code and interpret the output. Students may also need to fill in short lines of code.

- **HW**: Focus on putting together conceptual knowledge and computing skills. Most homework assingments include two parts: (1) *Concepts & Computations* - guided short-answer exercises that focus on conceptual knowledge and short computational tasks, (2) *Data Analysis* - open-ended question where students perform a complete regression analysis and write results as a narrative.
- **NOTES**: Supplemental notes providing more mathematical details. To fully understand the notes, the reader should be familiar with basic concepts in linear algebra.

1.2 Review: Intro Statistics and R

The primary audience for this text is students who have completed an introductory statistics course. It is assumed that students are familiar with the concept of statitical inference. This text is also written assuming students have had some exposure to R and the tidyverse syntax. (There is one "Intro to R" assingment included; however, this assignment is not a comprehensive introduction to R.) The following are suggested texts to review statistical concepts and computing:

- *OpenIntro Statistics*
- *Modern Dive*
- *R for Data Science*

Chapter 2

Notes for Instructors

This chapter includes details for setting up RStudio Cloud and GitHub in your course. The information in this chapter is based on the Infrastructure page in (dsb, 2018) and experience from previous courses.

2.1 Setting up your course

2.1.1 GitHub

GitHub can be used as a course management platform that is a more flexible alternative traditional platforms such as Blackboard and Sakai (Cetinkaya-Rudel and Rundel, 2018). It can be used as the central place to share course content and for students to submit assignments. It is structured to promote collaboration, so it is good for courses that include a lot of group assignments. It also gives students practice establishing a workflow that is more representative of what is currently being used in industry.

Before setting up your course, create an account on GitHub.com if you do not already have one. The chapter Register a GitHub account in (Bryan and Hester, 2019) has helpful guidelines to consider when making a GitHub username.

To set up your course organization in GitHub:

1. Create a new organization in GitHub. To keep things simple, you can name the organization based on the course number and semester. For example, the organization name for my most recent Regression Analysis course was STA210-Sp19.

2. Apply for the GitHub Education benefits to obtain free private repositories for your course organization. By default any repositories created in the course organization will be private and thus only visible and accessible to the student (or group of students if it's a team assignment) who owns the assignment and the instructor. This is to comply with Family Educational Rights and Privacy Act (FERPA).

3. Now you're ready to add students to the GitHub course organization. Have students create a GitHub username and send it to you. You can share the guidelines from (Bryan and Hester, 2019) to help students create usernames that are applicable beyond their time in your course that can be shared with future employers.

   I typically have students provide their GitHub username as part of the "Getting to Know You" survey at the beginning of the semester. It doesn't take long to create a GitHub username, so this could also be done as an activity on one of the first few days of class.

4. Add students to the GitHub course organization using the `ghclass` R package. The section "Adding students and creating teams" in Rundel and Cetinkaya-Rundel (2018) provides step-by-step instructions for adding new members to the course organization.

5. If students will be working in teams, you can add the teams to the GitHub course organization using by following the steps in (Rundel and Cetinkaya-Rundel, 2018). Any assignment repo for a group assignment will be visible and accessbile to all the members of the team and instructor.

6. You're done! Now you're ready to create your first assignment, which is described in the sections below.

2.1.2 RStudio Cloud

There are a few key benefits of using RStudio Cloud rather than having students install RStudio on their local machines:

- Students can get started working in RStudio immediately, since there is no need for students to install RStudio or configure it with Git.
- You can ensure students have the packages (and correct versions!) needed to complete assignments by installing them in the course workspace's base project.
- Students can copy any projects you share with everyone in the workspace, which makes in-class activities more feasible.
- As an instructor, you can view all student projects making it easier to help students remotely when coding issues occur.

To set up the workspace for your course:

1. Go to rstudio.cloud and log in or create a new acount. I used the option to log in using my GitHub credentials to keep things simple.

2. Click to create a "New Space" and follow the prompts to create a private workspace for your course.

3. By default, the workspace only holds 10 members and 25 projects. If you need more space for course, submit a request by emailing support@rstudio.com.

4. In the course workspace, go to the "Members" menu and click options. Change the access to "Shared" to create a sharing link that can be distributed to your students. Anyone who clicks the link will automatically be added to your course workspace. Students can log in to RStudio Cloud using their Github credentials.

   After the first few weeks of class, you can change the access to the course space to "Invitation required". At that point you would need to send an invitation to anyone else wanting to join the course workspace.

   See the RStudio Cloud guide for more details about adding members and specific member roles. Some suggested roles are

   - Primary instructor: Admin role that can manage membership, view, edit and manage all projects in the workspace

- Secondary instructors / Teaching assistants: Moderator role that can view, edit and manage all projects in the workspace
- Students: Contributor role (default) that can create, edit and manage their own projects.

4. You're done! Now you're ready to create your first project in RStudio Cloud.

2.2 Making Assignments

2.2.1 GitHub

1. Create a private assignment repo that contains any starter documents you wish to provide for the students. For example, the repo could contain the following:
   - R Markdown template for students to fill in their responses.
   - Any data required to complete the assignment in a `/data` sub folder.
   - README that includes link to assignment instructions.
2. Use the `ghclass` package to create individual or team student repos that are mirrors of the original assignment repo. See Creating a team assignment in (Rundel and Cetinkaya-Rundel, 2018) to step-by-step instructions.

   By default these repos will be private.
3. Students clone the repo into RStudio to complete the assignment.

2.2.2 RStudio Cloud

1. Create a new project in the course workspace. In the project, include any R Markdown templates, R scripts, data sets, and any other documents the students will need to complete the assignment or activity.

2. In the project options, click “Everyone” for “Who can view this project”. This will make the project visible to every member of the course workspace.

3. Students can make a copy of the project and can complete the assignment in their copy.

2.3 Additional Resources

- Tech Talk: Frictionless onboarding to data science with RStudio Cloud: information on using RStudio Cloud in your classroom.
- *Happy Git with R*: information on using Git, Github and RStudio.

Chapter 3

Intro to R

3.1 COMP: Intro to R

Introduction

The main goal of this assignment is to introduce you to R and RStudio, which we will be using throughout the course both to learn the statistical concepts discussed in the course and to analyze real data and come to informed conclusions. (*Note: R is the name of the programming language and RStudio is an interface.*)

An additional goal is to introduce you to git, a version control system, and GitHub, a collaboration system that is a place on the internet to host git-based projects. We will be using both of these tools throughout the seemster.

This assignment will focus on learning the basics of R and RStudio - understanding the interface, reading in data, and some basic commands. Future assignments will focus more specifically on regression analysis. This introduction to R and RStudio is not comprehensive. Students who are brand new to R should refer to {#review} for resources that provide a more comprehensive introduction to R.

This assignment should be completed **individually**, so you can get practice using git and GitHub before collaborating with others. You will work with others in future computing assignments.

Topics:

- Exploratory Data Analysis (data visualizations and numerical summaries)
- Simple linear regression
- Writing a report using R Markdown
- Tracking changes and submitting work using git and GitHub

Packages

We will use the following packages in today's assignment.

```r
library(tidyverse)
library(readr)
library(skimr)
library(broom)
```

If you need to install any of the packages, you can run the code below in the **console**.

```r
install.packages("tidyverse")
install.packages("readr")
install.packages("skimr")
install.packages("broom")
```

Warm up

YAML:

The top of the R Markdown file (between the three dashed lines) is called the YAML ("YAML Ain't Markup Language"). Basic information about your document, such as the title, author, output format, etc., is located in the YAML.

Open the R Markdown (.Rmd) file in your project, change the author name to your name, and knit the document.

Commiting changes:

Go to the **Git** pane in RStudio.

You should see the changes you made to the Rmd file listed here. Click to select the change (or changes) you made and click **Diff** to see the difference between the last committed state of the document and the current state that now includes your changes (in this case, writing your name in the author field). Now, write a short message in the **Commit message** box (e.g. a message like "Updated author name") and click **Commit**.

You don't have to commit after every change, but should commit states that are meaningful and may be helpful to use later for comparision. In the early assignments, there will be guidance to help you determine when to commit, and you will choose when to commit in later assignments.

Pushing changes:

Once you've committed a set of changes, the next step is to push them to the repo on GitHub.com. This is to ensure others can see your changes (in this class the instructor since your repos are only visible to you and the instructor). In other words, pushing changes is how you will submit your work for grading.

To push your changes to GitHub, click on **Push**. (*Note: If you are using RStudio Cloud, you may be asked to input your GitHub username and password. We will discuss how to save your GitHub credentials in a later assignment.*)

Data

Today's data comes from the Capital Bikeshare in Washington D.C (cap, 2019). The Capital Bikeshare is a system in which customers can rent a bike for little cost, ride it around the city, and return it to a station near their destination. We will read in the data from the file *bikeshare.csv* located in the *data* folder.

```r
bikeshare <- read_csv("data/bikeshare.csv")
```

This dataset contains the number of bike rentals, environmental conditions, and other information about the Capital Bikeshare everyday in 2011 and 2012.

Exercises

Before doing any analysis, we want to understand the basic structure of the data. One way to do this, is to look at the actual dataset. Type the code below in the **console** to view the entire dataset.

```r
View(bikeshare)
```

Because the dataset is large (i..e has a large number of observations), it is difficult to understand the data just by looking at the individual observations. We can use the `glimpse` function to view a summary of the data and get a general idea about the data structure. Using this function to view the data can be especially useful when importing data from a csv file (like in this assignment) to ensure that the imported dataset has the number of observations (rows) and variables (columns) we expect. We can also use this function to see the type for each of the imported variables (integer, character, etc.).

Note: To learn more about a function, type `??function` *in the console. For example, to learn more about the* `glimpse` *function, type* `??glimpse` *in the console.*

1. Type `glimpse(bikeshare)` in the **console** an overview of the `bikeshare` dataset.

How many observations are in the `bikeshare` dataset? How many variables?

2. In this assignment, we will focus the analysis on the following variables:

`season`	1: Winter, 2: Spring, 3: Summer, 4: Fall
`temp`	Temperature (in $^\circ C$) $\div$ 41
`count`	total number of bike rentals

Before fitting any regression models, we want to do an exploratory data analysis (EDA) to summarize the main characteristics of the data. Much of the EDA is visual, which we'll discuss more in the next exercise. The EDA also consists of calculating summary statistics for the variables in our dataset. It is good practice to examine any variable that may be relevant to the analysis in the EDA, since there may be variables that aren't directly included in the regression model but still provide important context to fully understand the results (i.e. lurking variables). To keep this assignment manageable, we will only examine the three variables `season`, `temp`, and `count`.

There are many ways to calculate summary statistics for each variable, and we will use a few of them throughout the semester. For now, let's use the `skim` function to calculate basic measures of center and spread and get a sketch of the distribution.

```
bikeshare %>%
  select(season,temp,count) %>%
  skim()
```

- What is the mean number of bike rentals?
- About 25% of the days in the data have a `count` above what value?

3. Does it make sense to calculate measures of center and spread for the variable `season`? If so, explain why it makes sense. Otherwise, explain why the `skim` function calculated these summary statistics for the variable `season` even if they don't make sense.

Knit and commit changes with the commit message "Added summary statistics (Ex 1 - 3)", and push.

Visualizing Your Data

4. One important part of EDA is visualizing the data to get a better understanding about the shape of the distribution for each variable and the relationship between variables. There are a lot of ways to make plots in R; we will use the functions available in the `ggplot2` package.

Note: https://ggplot2.tidyverse.org/ is a great resource as you learn `ggplot()`*. Click* ***Reference*** *in the top right corner to see a list of the various* *plot types available in the ggplot2 package.*

The code below is used to create a histogram to visualize the distribution of `count`. Modify the code by writing an informative title and label for the x-axis.

```
ggplot(data=bikeshare, mapping=aes(x=count)) +
  geom_histogram() +
  labs(title="______", x="______")
```

5. There may be times you want to customize a plot by changing features such as the color, marker types, etc. When plotting a histogram, one easy way to customize it is by changing the color of the bars. Let's look at two different ways to do this.

First, using a color of your choice, fill in the code below to include the option `color="_____"` inside of `geom_histogram()` function. Be sure to also include an informative title and label for the x-axis. You can use the ggplot2 quick reference or HTML color codes to help choose a color.

```
ggplot(data=bikeshare, mapping=aes(x=count)) +
  geom_histogram(color="_______") +
  labs(title="______", x="______")
```

Next, instead of `color="_____"`, use `fill="______"` inside of the `geom_histogram` code and fill in the color of your choice. It can be the same color you chose before or a different one.

What is the difference in the two plots? In other words, what is the difference in the way color displays on the histogram when using `color` versus `fill`?

6. Describe the distribution of `count`. Your description should include comments about the shape, center, spread, and any potential outliers. Refer to the histogram and the summary statistics from Exercise 2 in your description.

Knit and commit changes with the commit message "Added data visualization of count (Ex 4 - 6)", and push.

7. Now that we've examined the variables individually, we want to look at the relationship between the variables. To make interpretation easier, we will use the `mutate` function to create a new variable called `temp_c`

that is calculated as `temp * 41`. We will use `temp_c` for the remainder of the analysis, so the temperature can be discussed in terms of degrees Celsius.

```
bikeshare <- bikeshare %>%
  mutate(temp_c = temp * 41)
```

Complete the code below to make a scatterplot of the number of bike rentals versus the temperature (in degrees Celsius).

```
ggplot(data=bikeshare, mapping=aes(x=temp_c,y=count)) +
  ___________________
```

Describe the relationship between the temperature and the number of bike rentals.

8. The temperature and number of bike rentals varies greatly depending on the season. Therefore, we would like to create a separate scatterplot of `count` versus `temp_c` for each season. To do so, we will use the `facet_wrap` function, faceting by `season`. Recall from Exercise 2 that `season` is currently stored as an integer. We need to change it to a factor variable type before using it in the `facet_wrap` function.

Run the code below to see the replationship between temperature and number of bike rentals by season.

```
bikeshare <- bikeshare %>%
  mutate(season = as.factor(season))
```

```
ggplot(data=bikeshare, mapping=aes(x=temp_c,y=count)) +
  geom_point() +
  labs(title = "Number of Bike Rentals vs. Temperature",
       subtitle="Faceted by Season",
       x = "Temperature (Celsius)",
       y = "Number of Bike Rentals") +
  facet_wrap(~season)
```

For which season does the linear relationship between the temperature and the number of bike rentals appear to be the strongest?

Knit and commit changes with the commit message "Added visualization of count vs. temperature (Ex 7 - 8)", and push.

Simple Linear Regression

We want to fit a least-squares regression using the temperature (`temp_c`) to explain variation in the number of bike rentals (`count`) in the **winter** season. We can use the `filter` function to create a subset from the data that only includes days during the winter season. The `<-` assigns the name `winter_data` to our filtered subset.

```
winter_data <- bikeshare %>%
  filter(season=="1")
```

We will use `winter_data` for the remainder of the assignment.

9. Modify the code below to fit a simple linear regression model using the `lm` function; assign it the name `winter_model`. Replace X, Y, and *my.data* in the code below with the appropriate values.

```
winter_model <- lm(Y ~ X, data = my.data)
tidy(model) #output model
```

- Interpret the slope.
- Does it make sense to interpret the intercept? If so, write the interpretation of the intercept. Otherwise, explain why not.

10. We conclude by checking the assumptions for regression. We can use the `mutate` function to add a new variable called `resid` that is the residual for each observation in `winter_data`.

```
winter_data <- winter_data %>%
  mutate(resid = residuals(winter_model))
```

The code to plot the residuals vs. the predictor variable and make a Normal QQ-plot of residuals is shown below. In addition to these plots, write the code to make a histogram of the residuals. Refer to code from previous exercises to help you plot the histogram.

```
ggplot(data=winter_data, mapping=aes(x=temp_c,y=resid)) +
  geom_point() +
  geom_hline(yintercept=0,color="red")+
  labs(title="Residuals vs. Temperature",
       x="Temperature",
       y="Residuals")
```

```
ggplot(data=winter_data, mapping=aes(sample=resid)) +
  stat_qq() +
  stat_qq_line()+
  labs(title="Normal QQ Plot of Residuals")
```

- Based on the plots of the residuals and the scatterplot, are the linearity, normality, and constant variance assumptions met? Briefly explain your reasoning for each assumption.
- Is the independence assumption met? Briefly explain. You can use a description of the data and/or a plot to help you make this assessment.

Throughout the semester, we will learn various methods to deal with any violations in regression assumptions. For now, we will just note them.

Knit then commit all remaining changes, write a commit message indiciating you're finished, and push to GitHub. Before you finish, make sure all documents are updated on your GitHub repo.

Acknowledgement

This assignment was modeled on the "Hello R" lab from Data Science in a Box.

3.2 IN-CLASS: Movies on IMDB

In this activity, we will examine the relationship between budget and revenue for movies made in the United States in 1986 to 2016. The data is originally (imd, 2019).

```
#load packages
library(readr)
library(tidyverse)
library(DT)
```

Data

The dataset `movies.csv` includes basic information about each movie including budget, genre, movie studio, director, etc. A full list of the variables may be found here.

```
movies <- read_csv("https://raw.githubusercontent.com/danielgrijalva/movie
```

```
movies <- movies %>%
  filter(country=="USA", #use movies in the USA
         !(genre %in% c("Musical","War","Western"))) #removes genres with
```

```
#view the data
movies
```

Analysis

Part 1

We begin by observing how the gross revenue (`gross`) has changed over time. Since we want to visualize the results, we will choose a few genres of interest for the analysis.

```
genre_list <- c("Horror", "Drama", "Action", "Animation")
```

```
movies %>%
  filter(genre %in% genre_list) %>%
  group_by(genre,year) %>%
  summarise(avg_gross = mean(gross)) %>%
  ggplot(mapping = aes(x = year, y = avg_gross, color=genre)) +
    geom_line() +
    ylab("Average Gross Revenue (in US Dollars)") +
    ggtitle("Gross Revenue Over Time")
```

Part 2

Next, let's see the relationship between a movie's budget and its gross revenue. Because there is a large range of values for budget and revenue, we will plot the log-transformed version of each variable to more easily visualize the relationship. We will talk more about variable transformations later in the semester.

```r
movies %>%
  filter(genre %in% genre_list, budget > 0) %>%
  ggplot(mapping = aes(x=log(budget), y = log(gross), color=genre)) +
  geom_point() +
  geom_smooth(method="lm",se=FALSE) +
  xlab("Log-transformed Budget")+
  ylab("Log-transformed Gross Revenue") +
  facet_wrap(~ genre)
```

Next Steps

1. Put your name in the author field in the `YAML` at the top of the .Rmd file. Knit again. (We will talk more about the `YAML` later in future assingments.)
2. Change the genre names in parts 1 and 2 to genres that interest you. The spelling and capitalization must match what's in the data, so you can use the Appendix to see the correct spelling and capitalization. Knit again.

Discussion Questions

1. Consider the plot in Part 1.
   - Describe how movie revenue has changed over time.
   - Suppose we use revenue as a measure of popularity. How has the popularity of each genre changed over time? In other words, are the genres that were most popular in 1986 still the most popular today?
2. Consider the plot in Part 2.
   - Which genre(s) tend to have the highest budgets?
   - In general, what is the relationship between a movie's budget and its total revenue? Are there any genres that show a different relationship between budget and revenue?

Appendix

Below is a list of genres in the data set:

```r
movies %>%
  arrange(genre) %>%
  select(genre) %>%
  distinct() %>%
  datatable()
```

Chapter 4

Simple Linear Regression

4.1 Simple Linear Regression

4.1.1 Getting started

Putting text here

4.1.2 Foundation

Putting text here

4.1.3 Inference

Putting text here

4.2 Simple Linear Regression

4.2.1 Introduce data

Text goes here

4.2.2 Prediction

text goes here

4.2.3 Checking conditions

text goes here

4.2.4 Partioning variability

text goes here

4.3 Deriving β_1 and β_0

This document contains the mathematical details for deriving the least-squares estimates for slope (β_1) and intercept (β_0). We obtain the estimates, $\hat{\beta}_1$ and $\hat{\beta}_0$ by finding the values that minimize the sum of squared residuals (4.1).

$$SSR = \sum_{i=1}^{n}[y_i - \hat{y}_i]^2 = [y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)]^2 = [y_i - (\hat{\beta}_0 - \hat{\beta}_1 x_i]^2 \tag{4.1}$$

Recall that we can find the values of $\hat{\beta}_1$ and $\hat{\beta}_0$ that minimize (4.1) by taking the partial derivatives of (4.1) and setting them to 0. Thus, the values of $\hat{\beta}_1$ and $\hat{\beta}_0$ that minimize the respective partial derivative also minimize the sum of squared residuals. The partial derivatives are

$$\begin{aligned} \frac{\partial \text{SSR}}{\partial \hat{\beta}_1} &= -2\sum_{i=1}^{n} x_i(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) \\ \frac{\partial \text{SSR}}{\partial \hat{\beta}_0} &= -2\sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) \end{aligned} \tag{4.2}$$

Let's begin by deriving $\hat{\beta}_0$.

$$\begin{aligned}
\frac{\partial \text{SSR}}{\partial \hat{\beta}_0} &= -2\sum_{i=1}^{n}(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \\
&\Rightarrow -\sum_{i=1}^{n}(y_i + \hat{\beta}_0 + \hat{\beta}_1 x_i) = 0 \\
&\Rightarrow -\sum_{i=1}^{n} y_i + n\hat{\beta}_0 + \hat{\beta}_1 \sum_{i=1}^{n} x_i = 0 \\
&\Rightarrow n\hat{\beta}_0 = \sum_{i=1}^{n} y_i - \hat{\beta}_1 \sum_{i=1}^{n} x_i \\
&\Rightarrow \hat{\beta}_0 = \frac{1}{n}\Big(\sum_{i=1}^{n} y_i - \hat{\beta}_1 \sum_{i=1}^{n} x_i\Big) \\
&\Rightarrow \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}
\end{aligned} \tag{4.3}$$

Now, we can derive $\hat{\beta}_1$ using the $\hat{\beta}_0$ we just derived

$$
\begin{aligned}
\frac{\partial \text{SSR}}{\partial \hat{\beta}_1} &= -2\sum_{i=1}^{n} x_i(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) = 0 \\
&\Rightarrow -\sum_{i=1}^{n} x_i y_i + \hat{\beta}_0 \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 = 0 \\
(\text{Fill in } \hat{\beta}_0) &\Rightarrow -\sum_{i=1}^{n} x_i y_i + (\bar{y} - \hat{\beta}_1 \bar{x}) \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 = 0 \\
&\Rightarrow (\bar{y} - \hat{\beta}_1 \bar{x}) \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i \\
&\Rightarrow \bar{y} \sum_{i=1}^{n} x_i - \hat{\beta}_1 \bar{x} \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i \\
&\Rightarrow n\bar{y}\bar{x} - \hat{\beta}_1 n\bar{x}^2 + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i \\
&\Rightarrow \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 - \hat{\beta}_1 n\bar{x}^2 = \sum_{i=1}^{n} x_i y_i - n\bar{y}\bar{x} \\
&\Rightarrow \hat{\beta}_1 \Big(\sum_{i=1}^{n} x_i^2 - n\bar{x}^2 \Big) = \sum_{i=1}^{n} x_i y_i - n\bar{y}\bar{x} \\
\hat{\beta}_1 &= \frac{\sum_{i=1}^{n} x_i y_i - n\bar{y}\bar{x}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}
\end{aligned} \tag{4.4}
$$

To write $\hat{\beta}_1$ in a form that's more recognizable, we will use the following:

$$\sum x_i y_i - n\bar{y}\bar{x} = \sum (x - \bar{x})(y - \bar{y}) = (n-1)\text{Cov}(x, y) \tag{4.5}$$

$$\sum x_i^2 - n\bar{x}^2 - \sum (x-\bar{x})^2 = (n-1)s_x^2 \tag{4.6}$$

where $\mathrm{Cov}(x,y)$ is the covariance of x and y, and s_x^2 is the sample variance of x (s_x is the sample standard deviation).

Thus, applying (4.5) and (4.6), we have

$$\begin{aligned}
\hat{\beta}_1 &= \frac{\sum_{i=1}^{n} x_i y_i - n\bar{y}\bar{x}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2} \\
&= \frac{\sum_{i=1}^{n} (x-\bar{x})(y-\bar{y})}{\sum_{i=1}^{n} (x-\bar{x})^2} \\
&= \frac{(n-1)\mathrm{Cov}(x,y)}{(n-1)s_x^2} \\
&= \frac{\mathrm{Cov}(x,y)}{s_x^2}
\end{aligned} \tag{4.7}$$

The correlation between x and y is $r = \frac{\mathrm{Cov}(x,y)}{s_x s_y}$. Thus, $\mathrm{Cov}(x,y) = r s_x s_y$. Plugging this into (4.7), we have

$$\hat{\beta}_1 = \frac{\mathrm{Cov}(x,y)}{s_x^2} = r\frac{s_y s_x}{s_x^2} = r\frac{s_y}{s_x} \tag{4.8}$$

Chapter 5

Analysis of Variance

Chapter 6

Multiple Linear Regression

Chapter 7

Model Selection

Chapter 8

Logistic Regression

Chapter 9

Multinomial Logistic Regression

Chapter 10

Special Topics

Chapter 11

Data Sets

Below is a list of the datasets used in this book. More details about each dataset are coming soon.

- `advertising.csv`
- `airbnb_basic.csv`
- `airbnb_details.csv`
- `beer.csv`
- `bikeshare.csv`
- `evals_mod.csv`
- `fivethirtyeight-recent-grads.R`
- `framingham.csv`
- `gss2016.csv`
- `KingCountyHouses.csv`
- `movies.csv`
- `recent-grads.csv`
- `sesame.csv`
- `sis.csv`

- `spotify.csv`
- `test_songs.csv`

Bibliography

(2018). Data science in a box. https://datasciencebox.org/. Accessed: 2019-05-01.

(2019). Capital bikeshare. https://www.capitalbikeshare.com/.

(2019). Internet movie database. https://www.imdb.com/.

Bryan, J. and Hester, J. (2019). *Happy Git with R*.

Cetinkaya-Rudel, M. and Rundel, C. (2018). Infrastructure and tools for teaching computing throughout the statistical curriculum. *The American Statistician*, 72.

Rundel, C. and Cetinkaya-Rundel, M. (2018). *Tools for managing github class organization accounts*. R package version 0.1.0.