A desk reference for political data preprocessing and management

Damon C. Roberts

A desk reference for political data preprocessing and management

Damon C. Roberts

Last Compiled: 2023

Table of contents

List of Figures

Preface

This book was originally inspired by a blog post that I had written and tried to find a more formal outlet for. When writing the blog post, a nagging thought kept hanging in the back of my mind, "What if you write this as a book?" I knew that this was a dangerous thought. At the time I was preparing for the academic job market and was in the throes of writing my dissertation (another book project). What a dumb idea. But, I was passionate about the topic and needed something to do when I needed a break from my dissertation. So, this project was born.

1 Introduction

As I often teach my students in introductory quantitative research methods classes, each and every choice you make for a study has deep implications for the way that you interpret results, what results you come up with, and the conclusions you draw from them. This book is not going to focus on the particular estimator one may choose to model a binary outcome variable nor will it focus on best practices for interpreting those estimands. Rather, this book is meant to address the process that comes before that – how to clean (which is one step of data preprocessing) and manage one's data.

This is an often neglected part of the process. When we enter graduate school, often we learn about best practices for recoding variables, for dealing with missing data, and eventually learn that it is not a good idea to overwrite one's original `.csv` or `.dta` files with our cleaned data. While, I understand why this is the case, it is a concerning approach to quantitative research and for training those to engage in it.

As I am someone who has been tasked to bring, often reluctant, students along to learn the statistical programming language `R` as well as statistical concepts all the way through multiple regression in a 16-week timeframe, I am sympathetic to the tendency to just give students datasets that I've already cleaned and pre-processed. These challenges are even greater when one is tasked with teaching graduate students these things as there are high expectations for each student to master these topics by the end of the term so that they can be successful for the remainder of the program and their academic careers.

Though, I understand this position, it seems that we do not really grow out of these tendencies to learn how to pre-process our data in principled, as opposed to ad-hoc, ways. There are so many examples of replication materials that do not include any documentation (e.g., code) about how the researchers took their raw data and put it in an analyzable format. Their data goes from some messy spreadsheet and viola, it is now a super neat and tidy spreadsheet with variables constructed from the raw variables without any documentation about how that was done.

Where it is becoming harder and harder to find a published paper that has results that we can replicate, the lack of documentation for how the data came to be is disheartening. While many express shock at the fact that we only just now are discovering that university presidents and high-profile scholars who have had 30-year careers have been fabricating data (in a cynical view) or at least are making serious mistakes (in a more optimistic view) when processing their data, I am not surprised at all.

Concurrently, the standards for publishing a quantitative paper are increasing. We are expected to publish more papers that are of better quality. These requirements often force us to move quicker but somehow more accurately. One side effect of demands for quality is that we are often encountering datasets with more and more rows *and* columns in them. This produces more opportunities to make mistakes (to be less accurate) as well as to take longer to pre-process and manage the data since there is so much of it (less efficient).

There are three primary goals of this book. The first goal of this book is to convince others who handle, analyze, draw conclusions from, and make recommendations about policy and political outcomes to take a less ad-hoc and a more principled approach to managing and pre-processing their data. The second goal of the book is to offer recommendations about the ways that we can implement a more principled approach to data pre-processing and management. And the final goal of the book is to act as a useful desk reference for undergraduate students, graduate students, and researchers to the various options we have out there to document our data pre-processing and management – in terms of programming languages and particular libraries within it.

Addressing these three goals should help one be more efficient and more accurate in their data pre-processing and management. If one pre-processes and manages their data with code as opposed to ad-hoc, clicking around and editing a spreadsheet, this should provide some degree of readable documentation about how the analyzed data came to be. As I will discuss in the next chapter, the efficiency comes from choices about the programming languages and the particular libraries that one uses within that language that provide the functions to make this management or pre-processing easier.

1.1 Plan of the book

The following chapter is designed to provide an introduction to the different programming languages and libraries within those languages that are available to us for data pre-processing and management. In the chapter, I will provide directions to get set up with each programming languages, will offer a brief discussion of what libraries and functions are in these programming languages, will discuss some options has to choose from for managing these libraries and functions for each programming language, will discuss some of the most popular libraries used for data management and pre-processing in each of these languages, and will discuss the concept of efficient, readable, and replicable code. Throughout the chapter, I will compare and contrast how effective each coding language and their common data management and pre-processing libraries are for efficient, readable, and replicable code.

The third chapter starts our exploration of the common ways that we can think about managing our data. That is, "how do we keep track of our raw and cleaned data?" It will discuss common bare minimum requirements for data security for human subjects set by Institutional Review Boards in the United States, how a principled approach to data management can help increase one's data security practices, and will make an argument about a workflow that one should consider implementing when working with data throughout a given research project.

The fourth chapter continues the discussion about principled data management. In doing so, the chapter puts chapter three in practice. That is, we see how we can use code to achieve the lofty goals we set in chapter 3. In doing so, it provides coding examples for the data management tasks for each of the programming languages and the common libraries within each of those programming languages.

Now that we are familiar with the theoretical best-practices for data management as well as how to actually implement them, the fifth chapter will move into data pre-processing. This chapter in particular will focus on a type of data pre-processing often referred to as data cleaning or data munging. In doing so, the chapter will discuss common tasks that we have to perform, provide coding examples of how to do so, and will compare the accuracy and efficiency of each programming language and library for these tasks.

The sixth chapter will focus on a more advanced set of data munging steps – variable transformations, standardization and normalization, simple scale creation (e.g., creating an additive scale). Like the fifth chapter, the sixth chapter will provide coding

examples of how to perform these tasks and will compare the accuracy and efficiency of each programming language and their common libraries for these tasks.

The seventh chapter will focus on a special but important type of data that we often need to pre-process: missing data. Each programming language has a different way of internally documenting missing values for a variable. The chapter will discuss these common pitfalls, how to detect whether there is a missing value, and will discuss the need to engage in exploratory data analysis to determine the underlying pattern that creates the missing data and will introduce some common approaches to addressing them. In doing so, the chapter will provide code examples for the common libraries in each programming language. While doing so, it will also discuss the accuracy and the efficiency of these libraries and programming languages for these tasks.

References