Collecting and Cleaning Data

Therri Usher

Friday, February 06, 2015

Collecting Data

What Is So Important About Collecting and Cleaning Data?

1. No data, no analysis.
2. If you do not have the appropriate data, you may not be able to answer your research question.
3. The data typically has to be formatted so that you can perform an analysis using it.
4. The majority of the time spent on any data analysis is typically spent cleaning the data.

Quantitative and Qualitative Data

Most data can be broken into two groups: quantitative and qualitative.

Quantitative data

- Information about quantities
- Numerical data
- Ex: data generated from scientific experiments

Qualitative data

- Information expressed in a natural language
- Non-numerical data
- Ex: responses from interviews

Data Collection - Who?

Collected data can be separated into primary and secondary data.

Primary data

- "Original data that has been collected specifically for the purpose in mind" (Wikibooks)

Secondary data

- "Data that has been collected for another purpose" (Wikibooks)

Primary and Secondary Data

Primary data

- You can be sure that if collected correctly, the data will provide an answer to your research question.
- Collecting primary data is time-consuming and expensive.
- You can introduce bias in the way you collect the data, such as through asking questions and picking participants.

Secondary data

- Using secondary data is typically less time-consuming and expensive.
- Data is only as reliable as its collector, which is not you.
- You can introduce bias in choosing what parts of the data to use.

Data Collection - When?

Data can be collected either cross-sectionally or longitudinally.

Cross-sectional data

- Data is measured or collected from subjects at one single time point.
- Some studies measure at multiple time points, but not the same subjects.
- The general focus is between-subject comparisons.

Longitudinal data

- Data is measured or collected from subjects at multiple points in time.
- Subjects might be added to the study but the study still follows the same subjects until the study ends, subjects no longer participate, or subjects die.
- The focus is not only between-subject but also within-subject comparisons.

Cross-Sectional and Longitudinal Data

Cross-sectional data

- Cross-sectional data collection takes less time and money.
- Statistical models for cross-sectional data are typically simpler.
- You only have information about one time point so you cannot assess causality.
- You cannot compare within subjects.

Longitudinal data

- Longitudinal data allows you to infer about causality.
- You can compare between and within subjects.
- Statistical models are typically more involved.
- They typically require more money and a lot more time.

Data Collection - How?

There are various methods of data collection. For quantitative data, the common methods are

- Census
- Surveys
- Observational studies
- Experiments

Keep in mind that many data sources use more than one method, such as asking questions (surveying) in observational studies.

Comparison of Different Methods

Feasibility

- A census is typically not possible, particularly if the population of interest is large.
- It may not be feasible to conduct an experiment, which requires an intervention.
- Observational studies take as much work as experiments, depending on their scale, but can be possible in areas where experiments are not.

Comparison of Different Methods

Generalizibility

- A census is entirely generalizable – the entire population of interest has been assessed.
- Experiments are typically randomized, which makes a strong case for generalizability.
- In fact, unless there is randomization, generalizability may be an issue in the other methods, and can still be an issue if everyone does not have equal probability of being selected or if the analysis does not account for sampling methods.

Comparison of Different Methods

Causation

- Surveys and observational studies are not the best choices for inferring causality, but having longitudinal data can help strengthen causal predictions.
- Randomized experiments are the gold standard for causal inference.

Other Methods of Data Collection

- Registries
- Databases
- Social networks
- Websites

How Would I Use These Other Methods

First and foremost, if the data is non-public and controlled by someone else, you should always ask permission first. If the data is public, there are various tools such as SQL for databases.

R has packages that allow users to collect data from Facebook and Twitter, Rfacebook and TwitteR, respectively.

For websites, obtaining data that cannot be downloaded but is public requires the use of data scraping.

Data Scraping

Data scraping is a technique in which a computer program extracts data from human-readable output coming from another program.
-Wikipedia

Ways To Scrape Data

- Hire someone! (Please don't.)
- Use software built for data scraping.
- Write code in R.
  - XML package
  - scrapeR package
  - rjson package

Idea Behind Data Scraping

XML

- Each webpage has a page source, usually written in HTML or XML.
- Parse the webpage to obtain the HTML or XML code
- Get attributes and/or node values from the parsed code.

The thinking is similar for other text-formatting languages. Usually, a value is identified by a name or label. Parse the page and retrieve the values.

Example: CDC Vaccination Prices

I will obtain website links listed on the following website:
`http://www.cdc.gov/vaccines/programs/vfc/awardees/vaccine-management/price-list/archive.html`

I will also obtain data from the table on the following website:
`http://www.cdc.gov/vaccines/programs/vfc/awardees/vaccine-management/price-list/index.html`

A Word of Caution

The ethics of data scraping have not been fully detailed. Some websites might not appreciate someone scraping their website and might put up barriers in doing so. Some websites even discuss it in their Terms of Service.

My advice: **Never publish analyses based on data scraped from the internet without the website owner's permission.**

Active Learning Exercise #1

The class will be split into two groups. One group will be assigned to defend data scraping as a legitimate form of data collection. The other group will have to argue against data scraping. Use internet sources to form your argument. Be sure to document your sources. You will have to rely on mostly non-academic sources, such as blogs and news articles.

Next class, a representative from each group will "debate" (more like discuss) their viewpoints and we will create our own ethics of data scraping. I will give you a few minutes at the beginning of class to discuss your talking points with the rest of your group.

Active Learning Exercise #1

Helpful Hints:

- How does computational power play a role in data scraping?
  - Ex: one person writing one code versus entire teams using multiple computers and programs
- Who owns the data?
- What about the products that stem from data scraping, such as publications?
- Is data scraping actually worth it?

Active Learning Exercise #2 (Optional)

Statistics and sports is a rapidly growing area of statistics. The NFL, MLB, and NBA now employs many data analysts. At the intersection of sports and public health are the health risks to players in each sport.

Use ESPN to scrape data and obtain the names of every active player in the NBA. In addition, obtain links to every player's own ESPN profile.

Hint: Start off at `http://espn.go.com/nba/players`

What's The Big Deal About Data Scraping?

Even if you never scrape data again in your life, it is a reflection of data science itself.

1. It is still an ever-evolving field.
2. The ethics of it have not been fully formed.
3. There is no one correct way to do it.
4. You will sometimes have to debate and defend your choices.

Cleaning Data

From Raw to Processed Data

- At its rawest stage, data is messy.
  - Luckily, people are usually nice enough to do some processing before making it available.
- While this step of the analysis is probably the longest, there is little research on how to do it.
- There are varying frameworks and criterion for deeming a data set to be “clean”, “processed”, or “tidy”.

Benefits and Drawbacks of Cleaning Data

- Data analysis is hard, if not impossible, without it.
- It takes a lot of extra time in the beginning but can save you a lot of time in the long run.
- A lot of people clean their data now.

Tidy Data

It is an idea coined by Hadley Wickham.

Data is comprised of values and each value belongs to a variable and an observation. A variable is a measured attribute. An observation is a unit a variable is measured on.

Tidy data has the following three properties:

1. Each variable forms a column.
2. Each observation forms a row.
3. Each type of observational unit forms a table.

Other hallmarks are:

- Column names are the names of the variables.
- Variable names match across files.
- Everything is coded sensibly.

Raw, Technically Correct, and Consistent Data

There are three levels of data: raw, technically correct, and consistent.

Technically correct - A data set that is stored in a data frame with relevant column names and each column is of the class that adequately represents its values.

Consistent - Technically correct data that is fit for statistical analysis. Data in which missing values, special values, obvious errors, and outliers are either removed, corrected, or imputed.

Discussion Time

Discuss the two ideas of clean data with your group. What are the advantages and disadvantages of each? Is either one entirely correct? Have either changed your ideas about what clean data should be like?

Clean Data: My Opinion

Both have good points but neither one is entirely correct.

- I do not think each type of observational unit should form a table.
- I think the criterion for consistent data is a bit strict.

However, both are good starting points and highlight good properties of clean data: organized, properly formatted, and no impossible values.

How Messy is Messy Data?

A few examples:

- Values are the wrong format
- Strings have too many or too few spaces
- Dates get messed up
- Values get placed in the same column or row
- Data gets duplicated or worse, deleted

NA, NULL, NaN

- NA is a placeholder for missing values.
- NULL is similar to the null set. It does not have a length and does not have take space in a vector.
- NaN is the result of a calculation with an unknown result, but the result is known to not be a number

```
x <- c(1,2,NULL,4)
x
```

```
## [1] 1 2 4
```

```
sqrt(-1)
```

```
## Warning: NaNs produced
```

```
## [1] NaN
```

Useful Functions

match(x, y): Returns things in x that match y as their value and returns things that do not match as NA

```
match(c("cat","dog"), "a")
```

```
## [1] NA NA
```

```
match(c("cat","dog"), "cat")
```

```
## [1]  1 NA
```

Useful Functions

Do not forget the as functions:

- as.numeric()
- as.matrix()
- as.data.frame()
- etc. . .

```
num <- 1:10
as.character(num)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
```

Useful Functions

substr(x, start, stop): Returns only a substring of the string

```
substr(c("Monday", "Tuesday", "Wednesday"), 1, 3)
```

```
## [1] "Mon" "Tue" "Wed"
```

```
substr(c("Monday", "Tuesday", "Wednesday"), 4, 1000)
```

```
## [1] "day"    "sday"   "nesday"
```

Useful Functions

strsplit(x, split): Returns the string x separated at the phrase indicated by split

```
strsplit("Heart Disease", " ")
```

```
## [[1]]
## [1] "Heart"   "Disease"
```

```
strsplit("Heart Disease", "e")
```

```
## [[1]]
## [1] "H"        "art Dis" "as"
```

Useful Functions

str_trim(string, side): Returns the string with space removed from the indicated sides

```
require(stringr)
```

```
## Loading required package: stringr
```

```
str_trim("   Therri Usher ", side="both")
```

```
## [1] "Therri Usher"
```

```
str_trim("   Therri Usher ", side="left")
```

```
## [1] "Therri Usher "
```

```
str_trim("   Therri Usher ", side="right")
```

```
## [1] "   Therri Usher"
```

Useful Functions

grep(pattern, x): Returns indices for each string element in x that contains the pattern

NOTE: grepl does the same except returns TRUE or FALSE for each element

```
days <- c("Monday", "Tuesday", "Wednesday")
grep("day", days)
```

```
## [1] 1 2 3
```

```
grepl("day", days)
```

```
## [1] TRUE TRUE TRUE
```

Useful Functions

sub(pattern, replacement, x): Returns the string x with the first patterns replaced

NOTE: gsub replaces all patterns present

```
sub("e", "a", days)
```

```
## [1] "Monday"    "Tuasday"   "Wadnesday"
```

```
gsub("e", "a", days)
```

```
## [1] "Monday"    "Tuasday"   "Wadnasday"
```

Useful Functions

paste(. . . , sep): Pastes all the listed elements as a string with the indicated separator

```
paste("Mon", "day", sep="")
```

```
## [1] "Monday"
```

```
paste("Mon", "day", sep=" ")
```

```
## [1] "Mon day"
```

```
paste("Mon", "day", sep="@")
```

```
## [1] "Mon@day"
```

Useful Functions

tolower(), toupper(): Changes the case of the strings

```
tolower("UPPER")
```

```
## [1] "upper"
```

```
toupper("lower")
```

```
## [1] "LOWER"
```

Useful Functions

Sys.Date(): Returns today's date

date(): Returns the current date and time

```
Sys.Date()
```

```
## [1] "2015-02-09"
```

```
date()
```

```
## [1] "Mon Feb 09 19:47:16 2015"
```

Useful Functions

as.Date(x): Converts the string x to dates, represented as the number of days since January 1, 1970 (http://www.statmethods.net/input/dates.html)

```
dates <- as.Date(c("2015-02-12", "1989-02-19"))
days.lived <- dates[1] - dates[2]
days.lived
```

```
## Time difference of 9489 days
```

```
as.numeric(days.lived)
```

```
## [1] 9489
```

```
as.Date("31/1/2000", "%d/%m/%Y")
```

```
## [1] "2000-01-31"
```

Useful Functions

dmy(x): Formats the string x as a date

NOTE: If the order of day, month, and year is switched, use the proper function. For instance, if the order is month, day, year, then use the mdy() function.

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3
```

```
strings <- c("12/2/2015", "12-2-15", "12 February 2015")
dmy(strings)
```

```
## [1] "2015-02-12 UTC" "2015-02-12 UTC" "2015-02-12 UTC"
```

Useful R Packages

There are a couple of useful R packages for cleaning data:

- stringr package: string operations
- plyr package: manipulates and transforms data
- reshape2 package: aggregates data
- lubridate and Date: more date functions

My opinion: I am unsure of whether they are necessary.

Bottom Line

Cleaning data is tough. There's no standardized code that works for every data set. It takes thought and patience but it is worth it.

Active Learning Exercise #3

I have posted data on NBA players from the 2010-2011 NBA season. The data set contains their name, jersey number, position, date of birth, height, weight, college, country of citizenship, and years in the NBA. Separate the player names into two columns: first and last name. Change the position variable to a factor. Properly format the date of birth and add an age column. Make sure that the other variables are properly formatted as well.