Welcome to ML4CI

Ashley I Naimi

April 2023

Contents

1 Welcome to Machine Learning for Causal Inference

Welcome to ML4CI!

In this short course, we will spend the next few days learning about the basics behind using machine learning methods for estimating cause-effect relations.

By the end of this course you should have a solid understanding of:

- the challenges of estimating causal effects with data
- when/why one should use machine learning to estimate causal effects
- the challenges of using machine learning methods for estimating causal effects
- how to estimate causal effects with machine learning methods in R

The literature on ML and causal inference is very expansive and complex. There are some key essential concepts that are not easy to understand for those with little to no formal technical background. The purpose of this course is to help you with your efforts in filling this gap.

2 Overview of the Course

Over the next few days, we will cover the following topics:

- Introduction to the Datasets
- Potential Outcomes, Estimands, Identifiability
- Parametric Regression for Effect Estimation
  - G Computation
  - Inverse Probability Weighting
- Machine Learning for Effect Estimation: The Curse of Dimensionality
- Double Robust Methods: Some Intuition
  - Augmented Inverse Probability Weighting (AIPW)
  - Targeted Minimum Loss-Based Estimation (TMLE)
- Meta Learners for the Exposure and Outcome Models: Stacking
- SuperLearner and sl3
  - Tuning Parameter Grids

  - Selection Algorithms
- Estimating Effects in Example Datasets
  - TMLE3 + sl3 for the ATE, ATT, and ATU
  - AIPW + sl3 for the ATE, ATT, and ATU
- Machine Learning for Causal Effect Estimation: Wrapping Up
  - Alternative Estimands
  - Time-Dependent Exposure and Confounder Modeling
  - Mediation Analysis
  - Further Reading/Learning Materials

3 Some Logistics

All the materials for this short course are available in the following GitHub Repository: https://github.com/ainaimi/mlci_shortcourse

If you have git installed on your computer, you can download all the materials via the command line:

```
git clone https://github.com/ainaimi/mlci_shortcourse.git
```

Or you can use a GUI of your choice to clone the repo.

If you don't have `git` installed on your computer, you can simply download all the materials by visiting https://github.com/ainaimi/mlci_shortcourse.git, clicking the "clone" butting, and selecting "Download ZIP".

4 R Code Logistics

In this short course, we will be using the R programming language throughout. In order to run the code we will be using during the course of the workshop, it will be important to have a few packages pre-installed on your computer.

The easiest way to install these packages would be to paste the following code in the R console[1] and run it:

```
packages <- c("tidyverse", "here", "sandwich",
    "lmtest", "boot", "ranger", "ggplot2",
    "broom", "SuperLearner", "tmle", "AIPW",
```

[1] For the beginner, I highly recommend using RStudio as your IDE of choice, particularly if you are not familiar with R and R programming. There are many excellent resources for installing and setting up R and the RStudio IDE. Here is a good getting started guide, provided by Garrett Grolemund: https://rstudio-education.github.io/hopr/starting.html

```
    "ranger", "xgboost", "e1071", "nnet",
    "glmnet", "remotes")

for (package in packages) {
    if (!require(package, character.only = T,
        quietly = T)) {
        install.packages(package, repos = "http://lib.stat.cmu.edu/R/CRAN")
    }
}

for (package in packages) {
    library(package, character.only = T)
}

remotes::install_github("tlverse/tlverse")
library(tlverse)
```

If the above code does not work for you, consider installing each package separately:

```
install.packages("tidyverse")
library(tidyverse)
install.packages("here")
library(here)
install.packages("sandwich")
library(sandwich)
...
install.packages("remotes")
library(remotes)

remotes::install_github("tlverse/tlverse")
library(tmle3)
library(sl3)
```

4.1 Using `devtools` instead of `remotes`

Sometimes, installing development packages (i.e., using `install_github()`) can lead to installation errors and problems. If you encounter these errors, the best recommendation I can give is to try using `devtools` instead of `remotes`:

```
install.packages("devtools")
library(devtools)

devtools::install_github("tlverse/tlverse")
library(tmle3)
library(sl3)
```

4.2 Manually Installing Dependencies

If this doesn't work, the next option is to look carefully through the log at the error messages. Often, these errors arise because a certain dependency could not be installed. It would help to try to install those dependencies first, and then try again. This could mean sometimes installing compilation libraries on your computer first. For example,

`installation of package 'igraph' had non-zero exit status`

After running a successful `install.packages` for the `igraph` package, you could try installing `tlverse` again.

4.3 Addressing GitHub API Limits

There are two strategies one can pursue to deal with this error. First, the error arises because a single call to `install_github()` is attempting to install numerous packages at once. Without an authenticated API, this could easily reach the limit of request calls.

The easiest way to address this issue is to use a Github personal access token (PAT). There are a number of ways to do this, and it's important to read the basic information on PATs. Within R and RStudio, one straightforward way to manage PATs is to install and use the `usethis` package, which has a suite of functions available for creating and integrating PATs. Once you've installed `usethis`, you can:

- Use `usethis::browse_github_pat()` to create a GitHub token
- Use `usethis::edit_r_environ()` and add the environment variable by adding the following line to the R environment file: `GITHUB_PAT = 'your_github_token'`.
- Restart R (so that the GITHUB_PAT is read) and try to reinstall the packages that were resulting in the API limit error.

  Be aware: **your Github PAT is a password, and should be treated as such.**

4.4 Addressing Other Issues

There are many other reasons why errors in installing development packages such as the `tlverse` can arise. Too many to cover here. The final route you can take, if it is available to you, is to contact the IT person at your school or workplace. These folks usually know how to resolve cryptic errors, especially those due to missing compilers.

5 An Additional Note on Software

If all else fails, and you are unable to install the `tlverse`, which includes the `tmle3` and `sl3` packages, most of what we'll be doing in the course can be accomplished with two alternative packages that are available on CRAN: the `tmle` package by Susan Gruber (Gruber and van der Laan, 2012), and the `SuperLearner` package by Eric Polley (Polley et al., 2016).

These packages are easier to install (directly from CRAN in the traditional way), and easier to use. But they do not have as many options for implementing in different settings, and do not support the `pipeline` architecture the way the `tlverse` does `https://tlverse.org/tlverse-handbook/`.

References

Susan Gruber and Mark J van der Laan. tmle: An r package for targeted maximum likelihood estimation. *Journal of Statistical Software*, 51(13):1–35, 2012.

Eric Polley, Erin LeDell, Chris Kennedy, and Mark van der Laan. *SuperLearner: Super Learner Prediction*, 2016. URL https://github.com/ecpolley/SuperLearner. R package version 2.0-22.