CS 477/677 Causal Inference: Homework 1
Probability and Statistics
Due: Friday September 21, 2018, 11:59pm
50 Points Total

Ilya Shpitser
ilyas@cs.jhu.edu

1 Introduction

Homeworks in this class will consist of a programming part and a conceptual part.

1.1 Programming Assignments and the Background on the R Language

The programming part of the homework is meant to familiarize you with tools needed to perform causal inferences from data, and give you a chance to practice doing such analyses. This part will involve writing and running code, and perhaps answering some questions about the results. You will implement all assignments using the R programming language. Submissions in other languages will not be accepted. R is a very powerful high level programming language for data analysis and statistical computing. While no programming language is perfect, R's expressiveness and built in functions for common statistical operations will make writing data analysis programs simple compared to more "heavy duty" general purpose languages such as Java. R's semantics resemble the LISP programming language in many ways, while R's syntax resembles standard syntax used by the Algol family of languages (such as C, C++, Java, and python). If you are familiar with one of these languages, R will not be difficult for you to pick up and use. R can be downloaded for free at https://www.r-project.org/.

The R language has many predefined functions and a very large number of contributed packages for all sorts of things. To help you navigate, we will generally give suggestions for which functions or packages might be relevant. R is an interpreted programming language. You can evaluate code directly using a "read eval print loop" interpreter, load R scripts into an interpreter and run it there, or run R scripts from the command line using the command `Rscript` that comes with R. The most important R function for a beginner is `help(.)`. You can use it get a detailed description of any built in function in R, and generally on functions that are implemented in packages you download (provided those packages are well-documented!)

1.2 Conceptual Assignments and the Background on Latex

The conceptual part of the homework is meant to help you think through the philosophical and mathematical issues involved in causal inference. This part may involve manual calculation, mathematical and logical arguments, perhaps even essay questions. Your answers to the conceptual part of the homework must be submitted as a pdf file generated by latex. Latex is a very powerful typesetting system particularly useful for writing scientific and mathematical documents. While latex involves a bit of a learning curve compared to "what you see is what you get" editors such as Microsoft Word, the time investment is generally worth it. If you want to write scientific publications, latex is always a safe choice, as many journals and peer reviewed conferences have standardized on it. Latex can be viewed as a programming language that is compiled into a pdf file. A latex source file contains a combination of plain text and directives that instruct the compiler how to typeset the document. We recommend you use the command `pdflatex` to convert latex source code into pdf. This command is generally available on most common operating systems in use today. We will provide the latex source code for every homework assignment that we recommend you use as a standard point for the conceptual part of the assignment.

2 Programming (20 Points)

In this part of the assignment, you will familiarize yourself with the R language, and write functions for creating data frames (R data structures used to represent data sets), performing non-parametric bootstrap, constructing confidence intervals for any statistic of interest, and fitting parameters in a logistic regression model by solving estimating equations using the Newton-Raphson method. Unless otherwise stated, pleased implement all code yourself (even if an existing implementation already exists as a package, or even a part of standard R!)

2.1 Logistic Regression (10 Points)

You will implement fitting of a logistic regression model via solving estimating equations using a data set included with the homework. The following description of the dataset appears in [1], Chapter 8:

> This dataset is from the Job Search Intervention Study (JOBS II) [2]. In the JOBS II field experiment, 1,801 unemployed workers received a prescreening questionnaire and were then randomly assigned to treatment and control groups. Those in the treatment group participated in job-skills workshops. Those in the control condition received a booklet describing job-search tips. In follow-up interviews, two key outcome variables were measured: a continuous measure of depressive symptoms based on the Hopkins Symptom Checklist (`depress2`), and a binary variable representing whether the respondent had become employed (`work1`). In the JOBS II data, a continuous measure of job-search self-efficacy represents a key mediating variable (`job_seek`). In addition to the outcome and mediators, the JOBS II data also include the following list of baseline covariates that were measured prior to the administration of the treatment: pretreatment level of depression (`depress1`), education (`educ`), income, race (nonwhite), marital status (`marital`), age, sex, previous occupation (`occp`), and the level of economic hardship (`econ_hard`).

In this assignment, we will use a subset of the original dataset for which complete data on all features and outcome is available. We will discuss how to handle missing data later in the course. Your regression model will regress an intercept (represented by a column of 1s) and variables `treat, econ_hard, depress1, sex, age, occp, marital, nonwhite, educ, income,` and `job_seek` (the first 11 columns of the dataset) with the binarized outcome variable `outcome` (which will be added as the 17th column of the dataset). Your logistic regression function will take extra arguments, the third is a closure (more on closures below) representing a function $A(\mathbf{X})$, and the fourth is `tol`, which governs how close successive guesses of the parameters have to be before the Newton-Raphson loop exits. Your logistic regression function will solve, via Newton-Raphson, the estimating equations

$$\sum_{i=1}^{n} A(\mathbf{x}_{i(1:k)})\{y_i - \mu(\mathbf{x}_{i(1:k)}; \mathbf{w})\} = 0.$$

Your Newton-Raphson implemention ought to be able to handle any reasonable function $A(\mathbf{X})$. See question 2 in section 2.2.2 below.

Please do not use any built in R functions for fitting in this assignment, and implement fitting directly using matrix manipulation. You may find the matrix inversion function `solve(.)` and the matrix multiplication operator `%*%` helpful.

```
# Input:
#          X,y: a training dataset given as a set of feature rows, represented
#          by a n by k matrix X, and a set of corresponding
#          output predictions, represented by a n by 1 matrix y.
#          A: a function of the features x, used in estimating equations.
#          tol: tolerance used to exit the Newton-Raphson loop.
# Output:
#          A row vector of weights for a logistic regression model (with no intercept)
#          maximizing the likelihood of observing the data.
logisreg = function(X, y, A, tol = 0.01){

}
```

2.2 The Bootstrap

2.2.1 Implementation (6 Points)

You will implement the following four functions.

```
# Input:
#          mat: a matrix with n rows and k columns (rows are samples, columns are
#          variables).
# Output:  a data frame with column (variable) names "x1" to "xk", and data from the
#          matrix.
df_make = function(mat){

}
```

This function merely wraps a data frame around a matrix representing a data set, and gives standard names to columns (representing features or variables in our data). You might find functions `matrix(.)`, `as.data.frame(.)`, `dim(.)`, `paste(.)` and `colnames(.)` helpful.

```
# Input:
#          df: a data frame with n rows to be resampled.
# Output:  a data frame with n rows obtained from the input dataframe
#          by sampling rows with replacement.
df_resample = function(df){

}
```

This function constructs an appropriately resampled dataset used in nonparametric bootstrap. You might find the function `sample(.)`, and reading up on R's `1:k` syntactic sugar notation for generating vectors helpful.

```
# Input:
#          df: a data frame to be resampled
#          k: number of resampled datasets to generate.
#          f: a function of df giving the bootstrap statistic
#          (e.g. function(df) { mean(df$x1) })
#          q: a real number 0 < q < 0.5 giving the lower quantile of the desired
#          confidence interval.
# Output:  a four element vector giving the statistic of interest (first element),
#          and lower and upper confidence intervals corresponding to
#          q and 1-q quantiles (second and third elements) of the empirical
#          bootstrap distribution, and the size of the confidence interval.
bootstrap_ci = function(df, k, f, q){

}
```

This function uses non-parametric bootstrap to construct confidence intervals around a parameter of interest. We will use a particular rule which estimates this parameter using a data set. For example, if we were interested in the mean of the variable X_1, and we had data frame with a set of values for X_1, then we could use the maximum likelihood procedure for the mean. We will implement these rules by a *reified function*, sometimes known as a *closure* passed as the third argument. A closure that calculates the mean of X_1 from a data frame `df` could be constructed as follows:

```
f = function(df) { mean(df$x1) }
```

This creates a variable `f` in R which is a function of one argument, that expects a data frame, and which calculates and returns the mean of the `x1` column of this data frame. This variable can be treated as an ordinary variable and passed around to other functions, and it can be evaluated at any point by calling the variable with an argument: `f(df)`.

The function you will implement will first evaluate `f` on the original data set to obtain the statistic, generate `k` resampled data sets using the function `df_resample` above, store the values of the statistic of interest for each of these resampled datasets, and obtain the appropriate confidence intervals using the `quantile(.)` function applied to the difference of the original statistic, and the vector of statistics obtained from the resampled datasets, corresponding to the empirical bootstrap distribution. Remember, reported confidence intervals are centered around the statistic of interest. For example, if the statistic is an estimated mean of 3.2, the 0.05 quantile is -0.72, and the 0.95 quantile is 0.68, we report the vector $3.2, 2.48, 3.88, 1.4$.

You will use the following main function, please do not change it.

```
# Input:
#          none
# Output:
#          none
main = function(){

    # set the seed for the pseudo-random number generator.
    set.seed(0)
    # set the tolerance for Newton-Raphson.
    tol <- 0.01

    # load the dataset
    dat <- read.table("Jobs-NoMiss-Cont.tab", header = TRUE)

    # add a binarized version of depress2 called ‘outcome.’
    dat$outcome = 1 * (dat$depress2 >= 2)

    m <- as.matrix(dat)

    y <- m[,17, drop=FALSE]
    X <- m[,1:11]
    X <- cbind(rep(1, dim(X)[1]), X)

    A1 = function(x){
        x
    }

    w1 <- logisreg(X, y, A1, tol)

    print(w1)

    A2 = function(x){
        x^2
    }

    w2 <- logisreg(X, y, A2, tol)

    print(w2)

    k <- 3

    mu <- c(1, 2, 3)

    Sigma <- matrix(c(1, 1, 1, 1, 3, 1, 1, 1, 5), k, k)

    n <- 1000

    dat <- mvrnorm(n = n, mu, Sigma)

    df <- df_make(dat)

    mean_1 <- function(df) { mean(df$x1) }
    mean_2 <- function(df) { mean(df$x2) }

    k <- 1000

    print(bootstrap_ci(df, k, mean_1, 0.025))
    print(bootstrap_ci(df, k, mean_2, 0.025))
}
```

```
main()
```

Place this call at the end of your file such that running it as a script will evaluate the entry point function.

2.2.2 Questions (4 Points)

1 The bootstrap calls in `main()` give the confidence intervals for $\mathbb{E}[X_1]$ and $\mathbb{E}[X_2]$ where (X_1, X_2, X_3) are drawn from the following multivariate normal distribution:

$$\mathcal{N}\left((1,2,3), \begin{pmatrix} 1 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 5 \end{pmatrix}\right).$$

Out of the confidence intervals reported in `main()`, which confidence interval is wider, for the mean of X_1 or the mean of X_2? Why do you think that is?

2 Consider the vector function (of size $k \times 1$): $\sum_{i=1}^{n} A(\mathbf{x}_{i(1:k)})\{y_i - \mu(\mathbf{x}_{i(1:k)}; \mathbf{w})\}$. Compute the derivative of this function with respect to $\mathbf{w}$. Note that you should end up with a $k \times k$ matrix of derivatives.

3 Analytical (30 Points)

1 (4 points) Consider the following definitions of "actual causation" (one event causing another):

1 A causes B if the Pearson correlation coefficient ρ_{AB} is not equal to 0. The Pearson correlation coefficient is defined as

$$\rho_{AB} = \frac{\text{cov}(A, B)}{\sigma_A \sigma_B},$$

where σ_A is the standard deviation of A.

2 A causes B if $A \not\perp\!\!\!\perp B$ (meaning A is not marginally independent of B).

3 A causes B_t if $A \not\perp\!\!\!\perp B_t \mid B_{t-1}$ and A occurs prior to B_t in time. Here B_t and B_{t-1} are versions of a reoccurring variable B measured at time t and $t-1$.

4 A causes B if A occurs prior to B in time, and if A had not happened, B would not have happened.

For each definition, explain if it is reasonable. If not reasonable, write down a counterexample: a situation which obeys the definition but which intuitively does not correspond to what we mean when we say "A causes B."

2 (4 points) Show that the logistic regression model lies in the restricted moments model: $Y = \mu(\mathbf{X}; \beta) + \epsilon$, where $\mathbb{E}[\epsilon \mid \mathbf{X}] = 0$.

3 (3 points) Give three examples of *cargo cult behavior* in human beings: that is behavior based on an incorrect attribution of causation to an event. More on cargo cults can be found here:

https://en.wikipedia.org/wiki/Cargo_cult

Please don't use this example, obviously.

4 (4 points) Given an example of an inaccurate but precise estimator that gets lower MSE than an accurate but imprecise estimator.

5 (5 points) Show that the property $(\mathbf{A} \perp\!\!\!\perp_{\mathcal{G}} \mathbf{B} \mid \mathbf{C})$, meaning "in an undirected graph $\mathcal{G}$, all undirected paths from a vertex A in a set of vertices $\mathbf{A}$ to a vertex B in a set of vertices $\mathbf{B}$ is intersected by a subset of vertices in $\mathbf{C}$," satisfies the semi-graphoid axioms.

6 (6 points) Show that if $p(C, A, Y)$ is a distribution, then for any value a of A,

$$q_a(Y, C) = p(Y \mid A = a, C)p(C)$$

is also a distribution. That is, show $q_a(y, c) \geq 0$ for any y, c, and $\sum_{y,c} q_a(y, c) = 1$.

7 (4 points) Simpson's paradox occurs in joint distributions $p(Y, A, C)$ where

$$(\exists y, a_1, a_2) \text{ s.t. } p(y|a_1) < p(y|a_2), \text{ but } (\forall c)\ p(y|a_1, c) \geq p(y|a_2, c). \tag{1}$$

Define $q_a(Y, C)$ as in question 1, and let $q_a(Y) = \sum_C q_a(Y, C)$, $q_a(Y \mid C) = q_a(Y, C)/q_a(C)$. Show that the negation of (1) holds for q_a. In other words, show that

$$(\forall y, a_1, a_2)[(\forall c)\ q_{a_1}(y \mid c) \geq q_{a_2}(y \mid c)] \Rightarrow [q_{a_1}(y) \geq q_{a_2}(y)].$$

4 How to Submit

You can obtain the assignment from the class Piazza page. You will submit the assignment on Gradescope. There will generally be no extensions to the listed deadline.

5 What to Submit

A pdf file named homework1.pdf, and an R script named homework1.R. Please include your name, and email at the top of both files.

6 Questions?

You can ask questions about the homework, or any other aspect of the class at the class piazza page, found here:

http://piazza.com/jhu/fall2018/cs600477677/home

To signup for piazza for the class, follow this link:

http://piazza.com/jhu/fall2018/cs600477677

References

[1] Hrishikesh D. Vinod. *Advances in Social Science Research Using R.* Springer-Verlag New York, 2010.

[2] A. Vinokur and Y. Schul. Mastery and inoculation against setbacks as active ingredients in the jobs intervention for the unemployed. *Journal of Consulting and Clinical Psychology*, 65(5):867–877, 1997.