Economics 628 : Assignment 1

Paul Schrimpf
Due: September 12th, 2018
University of British Columbia
Economics 326

1. Theory questions

Problem 1: In the matching model from section 1.2 of the notes, let $p_0(x) = \mathrm{P}(d = 1|x)$ and $\mu(a, x) = \mathrm{E}[y|d = a, x]$.

(1) Let

$$\begin{aligned}\theta^a =& \mathrm{E}\left[\frac{y_i d_i}{p(x_i)} - \frac{y_i(1-d_i)}{1-p(x_i)}\right]\\ \theta^b =& \mathrm{E}\left[\mu(1,x_i) - \mu(0,x_i)\right]\\ \theta^c =& \mathrm{E}\left[d_i\frac{y_i - \mu(1,x_i)}{p(x_i)} - (1-d_i)\frac{y_i-\mu(0,x_i)}{1-p(x_i)} + \mu(1,x_i) - \mu(0,x_i)\right]\end{aligned}$$

Show that $\theta^a = \theta^b = \theta^c = \mathrm{E}[y(1) - y(0)]$

(2) Following the notation from section 1.6, let

$$\begin{aligned}\psi^a(w_i;\theta,p,\mu) =& \theta - \left[\frac{y_i d_i}{p(x_i)} + \frac{y_i(1-d_i)}{1-p(x_i)}\right]\\ \psi^b(w_i;\theta,p,\mu) =& \theta - \left[\mu(1,x_i) - \mu(0,x_i)\right]\\ \psi^c(w_i;\theta,p,\mu) =& \theta - \left[d_i\frac{y_i-\mu(1,x_i)}{p(x_i)} + (1-d_i)\frac{y_i-\mu(0,x_i)}{1-p(x_i)} + \mu(1,x_i) - \mu(0,x_i)\right]\end{aligned}$$

Calculate the Gâteaux (directional) derivatives of $\mathrm{E}[\psi^a(w_i;\theta,p,\mu)]$ with respect to p and μ at p_0, μ_0. Do the same for ψ^b and ψ^c.

Problem 2: Consider the following IV model:

$$\begin{aligned}y_i &= \theta d_i + \epsilon_i\\ d_i &= f(x_i) + u_i\end{aligned}$$

with $\mathrm{E}[\epsilon|x] = 0$ and $\mathrm{E}[u|x] = 0$. Let $\hat{f}()$ be an estimate of f that satisfies the following high level assumptions:

$$\mathbb{E}_n[d_i\hat{f}(x_i)] - \mathrm{E}[d_i f(x_i)] = O_p(n^{-1/2} + r_n) \tag{1}$$

$$\mathbb{E}_n[(\hat{f}(x_i) - f(x_i))\epsilon_i] = O_p(n^{-1/2} r_n) \tag{2}$$

for some $r_n \to 0$. Show that the estimate

$$\hat{\theta} = \mathbb{E}_n[\hat{f}(x_i)d_i]^{-1}\mathbb{E}_n[\hat{f}(x_i)y_i]$$

is $\sqrt{n}$ asymptotically normal. State any additional assumptions needed. *Hint*: *follow the steps of section* 1.6.1 *in the notes.*

2. Empirical questions

Problem 3: `http://tryr.codeschool.com/` is an interactive introduction to R. Please work through it if you have not used R before. If you're already familiar with R, then you can skip this.

If you dislike tryR, alternative introductory resources include chapter 2.3 of James et al. (2013) and https://swirlstats.com/students.html.

If you're new to R, here is some advice about additional tools to use. You can download R from CRAN.[1] R itself comes with either no GUI nor text-editor (on Linux) or a basic GUI with a limited text editor, (on Windows and I'd guess on Mac, but I don't know). There are numerous programs that provide a nicer way of working with R. The most popular is RStudio. It gives nice syntax highlighting, easier debugging, etc; it is somewhat similar to Matlab's GUI.

A potentially useful, but not essential, tool for writing assignments is the Rmarkdown package. It lets you combine R code and text into a single document and produces nice looking output in multiple formats. I'm using Rmarkdown to create the slides and notes for this portion of the course. For research, I often use Rmarkdown for preliminary data work that I'm just looking at or sharing with coauthors and still making changes frequently.

Problem 4: Explore the predictive performance of machine learning in your favorite dataset. Follow steps similar to the examples in section 2.1 of the notes. Fit OLS, Lasso, random forest, and optionally additional machine learning estimators using a randomly chosen training subset of your data. Then create table(s) and/or figure(s) comparing the performance of the estimators.

Suggested steps:

(1) **Load the data and needed packages:**

If your favorite dataset happens to already be in R format you can load it into R with simply,

```
load("favoriteData.Rdata")
ls() # print all objects in current environment, to find out the
     # name of the dataframe and other object in "favoriteData.Rdata"
```

More likely, your dataset will be in another format, perhaps csv

```
data <- read.csv("favoriteData.csv") # you may need to add some options
                                     # enter "?read.csv" for details
```

or Stata,

```
library(foreign)  # you will have to install this package with
                  # 'install.packages("foreign")' first
data <- read.dta("favoriteData.dta")
```

If you don't have a favorite dataset, R and many R packages include some datasets. For example, some included in R are listed here. Most of those are rather small and not the best choice for this assigment. The "hdm" package, which we will use later, contains some larger datasets from various economics papers, Chernozhukov, Hansen, and Spindler (2016). To use them:

```
install.packages("hdm") # install the package
library(hdm)
data(AJR) # load Acemoglu, Johnson, & Robinson data (or BLP,
          #         EminentDomain, GrowthData, or pension)
summary(AJR)
```

[1] You might also consider the version of R distribution by Microsoft, Microsoft R Open. The main benefit is that Microsoft R Open uses Intel's Math Kernel Library for linear algebra, which is quite fast. It's also possible to use Intel's MKL or another high performance BLAS & Lapack implementation (like openBLAS) witah the plain version of R and get similar performance.

Journal websites are another potential source of data. Textbooks, like Efron and Hastie (2016), often include datasets.

You should check that your data has been read correctly and contains what you expect by looking at some summary statistics

```
summary(data)
```

We'll use the "glmnet" package for Lasso and "grf" for random forests. We might as well as also install "ggplot2" for plotting later. Install them with

```
install.packages(c("glmnet","grf","ggplot2"))

# load them
library(glmnet)
library(grf)
library(ggplot2)
```

(2) **OLS:** Suppose you have a variable named "outcome" that you want to predict with variables names "x1", "x2", and "x3". You can do the following

```
# missing values will create size mismatches later. there are other
# workarounds (see na.action), but let's just drop them
data <- subset(data, rowSums(is.na(data[,c("outcome","x1","x2","x3")]))==0)

train <- runif(nrow(data))<0.5 # use half data for training

# estimate ols on training sample
ols <- lm(outcome ~ x1 + x2 + x3, data=subset(data, train),
          x=TRUE, y=TRUE) # we want the output to include the x and y
                          # matrices
data$y.hat.ols <- predict(ols, newdata=data) # make predictions for whole
                                             # sample
# compute mse and mae for training and holdout samples
by(data, train, FUN=function(df) {
  err <- with(df, y.hat.ols-outcome)
  out <- c(mean(err), mean(err^2), mean(abs(err)))
  names(out) <- c("mean error","MSE","MAE")
  return(out)
})
```

(3) **Random forest:** to estimate a random forest and get predictions:

```
Xt <- ols$x[,2:ncol(ols$x)] # we don't want the column of 1's
yt <- ols$y
rf <- regression_forest(Xt,yt,tune.parameters=TRUE)
# for prediction, we need X for the whole sample, so
olsall <- lm(outcome ~ x1 + x2 + x3, data=data, x=TRUE)
X <- olsall$x[,2:ncol(olsall$x)]
y.hat.rf <- predict(rf, X)$predictions
```

You can then calculate MSE and other statistics the same as in the OLS example.

(4) **Lasso:** to estimate a Lasso model and get predictions, you should first expand your X matrix to have more variables. If you data includes many more potential regressors, you can just include them. If not, you can add interactions, powers, and other transformations of the original regressors. The following includes a 2nd degree polynomial in the 3 regressors and a linear spline in each with knots at -1, 0, and 1.[2]

[2]This choice is completely arbitrary and is not meant as a good suggestion. Instead, it is meant to illustrute some of the features of R's formulas. See the code for the pipeline example in the notes for a more practical specification.

```
bigreg <- lm(outcome ~ polym(x1, x2, x3,degree=2) +
               (I(x1>-1) + I(x2>-1) + I(x3>-1) +
                I(x1>0) + I(x2>0) + I(x3>0) +
                I(x1>1) + I(x2>1) + I(x3>1))*(x1+x2+x3),
             data=data)
Xlasso <- Xlasso <- bigreg$x[,2:ncol(bigreg$x)]
lasso <- cv.glmnet(Xlasso[train, ], yt, alpha=1,
                   standardize=TRUE, intercept=TRUE)
data$y.hat.lasso <- predict(lasso, Xlasso, s=lasso$lambda.min,
                            type="response")
```

(5) If you want, you could create plots of the densities of errors or the prediction vs actual outcome, as in the notes.

References

Chernozhukov, Victor, Chris Hansen, and Martin Spindler. 2016. "hdm: High-Dimensional Metrics." *R Journal* 8 (2):185–199. URL https://journal.r-project.org/archive/2016/RJ-2016-040/index.html.

Efron, Bradley and Trevor Hastie. 2016. *Computer age statistical inference*, vol. 5. Cambridge University Press. URL https://web.stanford.edu/~hastie/CASI/.

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning*, vol. 112. Springer. URL http://www-bcf.usc.edu/%7Egareth/ISL/.