Chad Germany, Xinhao Liu
Stat 480: Big Data Analytics
October 14, 2023

Homework 3

Predicting House Prices.

Question 1. Id transform degree to create our treatment variable d. What would you do and why?

The degree tells you the connectivity level. The higher the connectivity the higher the treatment. The histogram of degree has a high skew towards zero this will negatively affect the model. So we can do a log(degree +1) to fix this and make it more like a gaussian distribution.

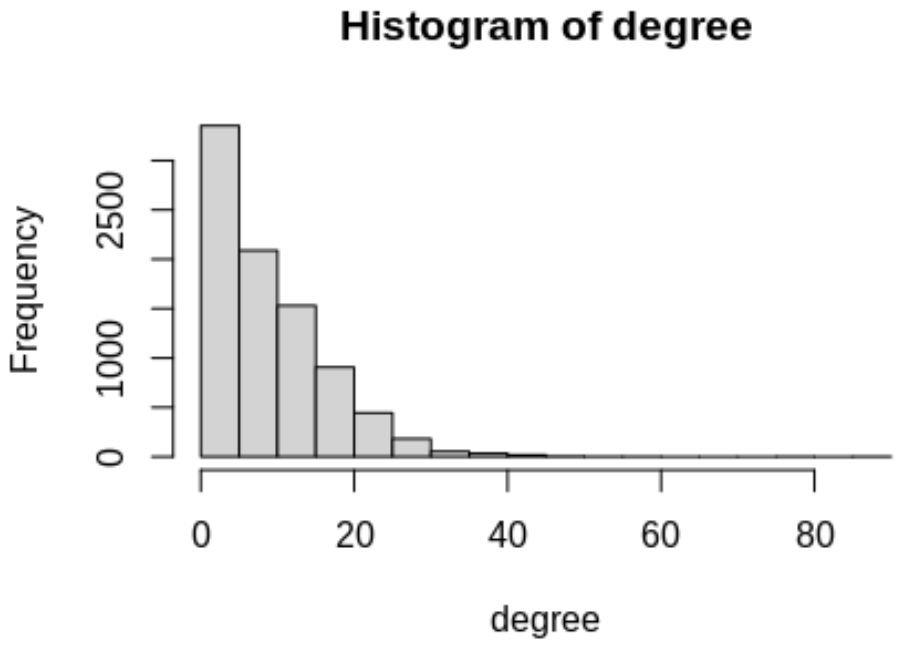

FIGURE 1. The histogram of degree. The higher the degree the more connected the village. We see a screwed distribution.

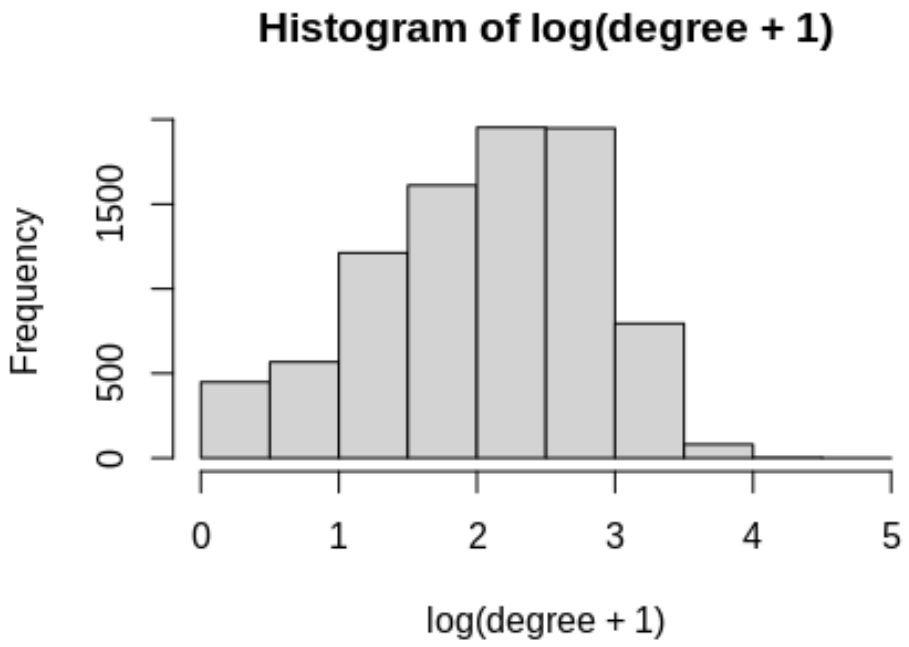

FIGURE 2. The histogram of degree. The distribution is more centered like a normal distribution.

Question 2. Build a model to predict d from x, our controls. Comment on how tight the fit is, and what that implies for estimation of a treatment effect.

If you just run a straight lasso onto log(degree+1) and x, AICc selects a positive value coefficient of .1171. The R^2 value is .0211 which tells you the fit. The function used "gamlr" selects the optimal lambda and gives us the controls for free.

Question 3. Use predictions from Q2 in an estimator for the effect of d on loan.

Doing the treatment effects LASSO we do two stages: Firstly estimate dhat(x) with lasso regression of d on x and secondly do a lasso of y on [d,dhat(x), x] with dhat(x) unpenalized

Doing a double lasso we were able to find a coefficient of .1075 which is a decrease from the previous result. The R^2 value is .0182 which is also a slight decrease from the result.

Question 4. Compare the results from Q3 to those from a straight (naive) lasso for loan on d and x. Explain why they are similar or different.

As we explained earlier there is a slight decrease in values for both R^2 and the coefficients. Looking at the graph dflog and dhat we see there is a slight correlation between the two. The R^2 value between the two is 0.0705 so we can say they are mostly independent of each other.

Question 5. Bootstrap your estimator from Q3 and describe the uncertainty. With bootstrapping you are pretending that the empirical data distribution is the population, and using it to draw alternative samples. Bootstraps see how varies for random draws from the joint distribution for [x, y]. MLE standard errors measure variation in for random draws from the conditional distribution [y |x]. In figure 4 we can see there is distribution similar to normal distribution. Minimum = 0.09652, the 1st Quartile = 0.11907, Median = 0.14168, 3rd Quartile = 0.16283 , Maximum = 0.20760

```
## microfinance network
## data from BANERJEE, CHANDRASEKHAR, DUFLO, JACKSON 2012

```

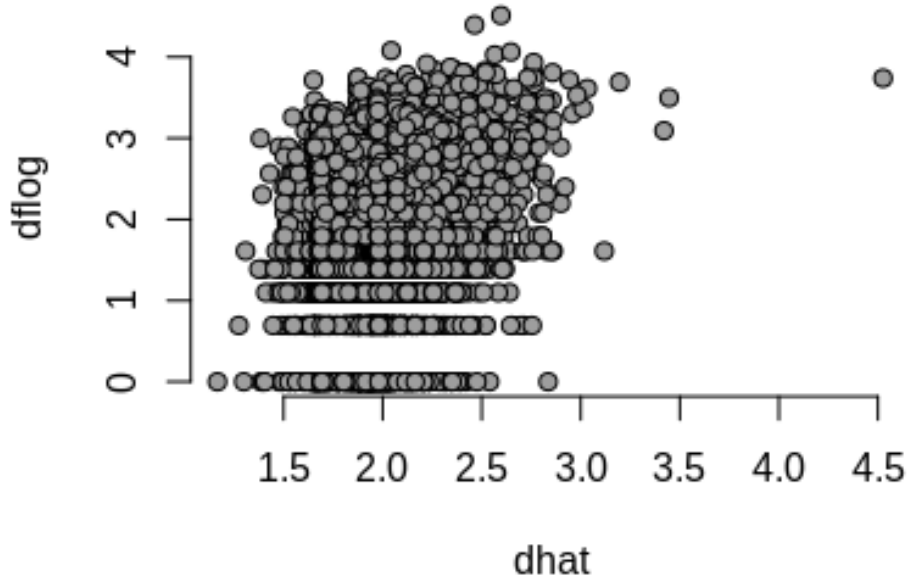

FIGURE 3. Scatterplot of dflog versus dhat. Shows how minimal the correlation between dflog and dhat is.

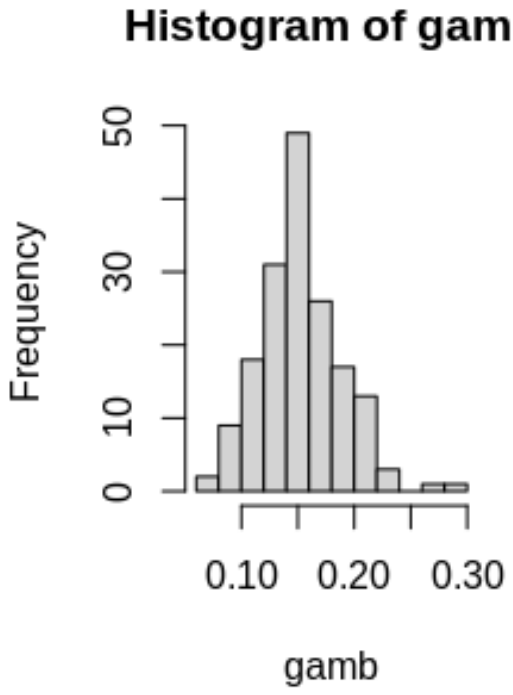

FIGURE 4. Bootstrap histogram of our double lasso treatment effect estimator. The distribution looks similar to normal distribution.

```
4
5  ## data on 8622 households
6  hh <- read.csv("microfi_households.csv", row.names="hh")
7  hh$village <- factor(hh$village)
8
9  ## We'll kick off with a bunch of network stuff.
10 ## This will be covered in more detail in lecture 6.
11 ## get igraph off of CRAN if you don't have it
12 install.packages("igraph")
13 ## this is a tool for network analysis
14 ## (see http://igraph.sourceforge.net/)
15 library(igraph)
16 edges <- read.table("microfi_edges.txt", colClasses="character")
17 ## edges holds connections between the household ids
18 hhnet <- graph.edgelist(as.matrix(edges))
19 hhnet <- as.undirected(hhnet) # two-way connections.
20
21 ## igraph is all about plotting.
22 V(hhnet) ## our 8000+ household vertices
23 ## Each vertex (node) has some attributes, and we can add more.
24 V(hhnet)$village <- as.character(hh[V(hhnet),'village'])
25 ## we'll color them by village membership
26 vilcol <- rainbow(nlevels(hh$village))
27 names(vilcol) <- levels(hh$village)
28 V(hhnet)$color = vilcol[V(hhnet)$village]
29 ## drop HH labels from plot
30 V(hhnet)$label=NA
31
32 # graph plots try to force distances proportional to connectivity
33 # imagine nodes connected by elastic bands that you are pulling apart
34 # The graphs can take a very long time, but I've found
35 # edge.curved=FALSE speeds things up a lot.  Not sure why.
36
37 ## we'll use induced.subgraph and plot a couple villages
38 village1 <- induced.subgraph(hhnet, v=which(V(hhnet)$village=="1"))
39 village33 <- induced.subgraph(hhnet, v=which(V(hhnet)$village=="33"))
40
41 # vertex.size=3 is small.  default is 15
42 plot(village1, vertex.size=3, edge.curved=FALSE)
43 plot(village33, vertex.size=3, edge.curved=FALSE)
44
45 ######  now, on to your homework stuff
46
47 library(gamlr)
48
49 ## match id's; I call these 'zebras' because they are like crosswalks
50 zebra <- match(rownames(hh), V(hhnet)$name)
51
52 ## calculate the `degree' of each hh:
53 ##  number of commerce/friend/family connections
54
55
56 ## if you run a full glm, it takes forever and is an overfit mess
57
58 # Warning messages:
59 # 1: glm.fit: algorithm did not converge
60 # 2: glm.fit: fitted probabilities numerically 0 or 1 occurred
61
62 #summary(full <- gamlr(loan ~ dflog + .^2, data=hh, family="gaussian"))
63
```

```r
hist(degree)
hist(log(degree+1))

degree <- degree(hhnet)[zebra]
names(degree) <- rownames(hh)
degree[is.na(degree)] <- 0 # unconnected houses, not in our graph
#q2

#exclude loan from hh
hh_sub = hh[,-1]
#do naive way
x = sparse.model.matrix(~  .^2, data=hh_sub)[,-1]
naive <- gamlr(cbind(dflog,x),hh$loan,family = "binomial")
coef(naive)["dflog",] #
summary(naive)[which(summary(naive)[,5] == min(summary(naive)[,5])),]

#q3
summary(full <- glm(loan ~ degree + .^2, data=hh, family="binomial"))

dflog = log(degree + 1 )
# summary(full <- glm(loan ~ dflog + .^2, data=hh, family="binomial"))
# d_hat = predict(full, data=hh, type="response")
# summary(full2 <- glm(loan ~ dflog+ d_hat + .^2, data=hh, family="binomial"))
# coef(full2)["dflog"]


treat <- gamlr(x,dflog,lambda.min.ratio=1e-4)
plot(treat) # there are some x's predictive of trestment

dhat <- predict(treat, x, type="response")
causal <- gamlr(cbind(dflog,dhat,x),hh$loan,family = "binomial", free=2,lmr=1e-4)
plot(causal)
coef(causal)["dflog",]

summary(causal)[which(summary(causal)[,5] == min(summary(causal)[,5])),]
#q4
plot(dhat,dflog,bty="n",pch=21,bg=8)
## IS R^2?
## Note: IS R2 is what governs how much independent signal
## you have for estimating
R2 = cor(drop(dhat),dflog)^2

# summary(full <- glm(loan ~ dflog + .^2, data=hh, family="binomial")) #naive lasso
# coef(full)["dflog"]



#q5
y <- hh$loan
n <- nrow(x)

## Bootstrapping our lasso causal estimator is easy

gamb <- c() # empty gamma

for(b in 1:100){
  ## create a matrix of resampled indices

  ib <- sample(1:n, n, replace=TRUE)

```

```r
  ## create the resampled data

  xb <- x[ib,]

  db <- dflog[ib]

  yb <- y[ib]

  ## run the treatment regression

  treatb <- gamlr(xb,db,lambda.min.ratio=1e-3)

  dhatb <- predict(treatb, xb, type="response")

  fitb <- gamlr(cbind(db,dhatb,xb),yb, family = "binomial",free=2)

  gamb <- c(gamb,coef(fitb)["db",])

  print(b)
}

## not very exciting though: all zeros
summary(gamb)
hist(gamb)
abline(v=coef(fitb)["db"],col=2)
```

LISTING 1. R source code implemented for the homework.