SCRIPT MOD5S2: TREATMENT EFFECTS VIA PROPENSITY SORE WEIGHTING: JOB TRAINING APPLICATION

1. Load and describe data

This script uses the same data as `mod5s1`.

(1) train =1 if assigned to job training
(2) age =age in 1977
(3) educ =years of education
(4) black =1 if black
(5) hisp =1 if Hispanic
(6) married =1 if married
(7) nodegree =1 if no high school degree
(8) mosinex =No. months prior to Jan. 78 in experiment
(9) re74 =real earnings, 1974, \$1000s
(10) re75 =real earnings, 1975, \$1000s
(11) re78 =real earnings, 1978, \$1000s
(12) unem74 =1 if unemployed all of 1974
(13) unem75 =1 if unemployed all of 1975
(14) unem78 =1 if unemployed all of 1978
(15) lre74 =log(re74); zero if re74 == 0
(16) lre75 =log(re75); zero if re75 == 0
(17) lre78 =log(re78); zero if re78 == 0
(18) agesq =age squared
(19) mostrn =months in training

Our goal will be to replicate the results reported in Wooldridge (2010), Table 21.1, p. 929, for the "jtrain2" sample. Because of rounding errors and (I suspect) subtle differences in STATA's and R's "inverse" functions, our standard errors will be a bit different from those given in Wooldridge's table.

The following loads in the data (including all variable names):

```
R> load("c:/Klaus/AAEC5126/R/data/jtrain2.rda")
```

2. Computing propensity scores and checking for overlap

The estimation of the PS for each observation requires a binary probability model, such as logit, probit, or nonparametric methods. Here we follow Wooldridge (2010) and use a simple logit equation. We could code up a logit MLE problem from scratch, or simply use `R` 's built-in "glm" (generalized linear models) function. Since the focus of this script is not on MLE techniques, we'll go with the latter.

```
R> attach(data)
R> n<-nrow(data)
```

```
R> n1<-185
R> n0<-n-n1 #260 cases
R> y<-train
R> X<-cbind(age,educ,black,hisp,married,re74,re75)
R> #
R> logit.out<-glm(train~age+educ+black+hisp+married+re74+re75,data=data,family=binomial)
R> # note: R automatically adds a constant term,
R> # so don't include a colum of 1's in X
R> # we're not particularly interested in the estimated marginal effects,
R> # so we'll jump right to the propensity scores
R> PSvec<-predict(logit.out,newdata=NULL,type="response")
R> # n by 1 vector of fitted probabilities, aka "propensity score"
R> #
R> #Checking for overlap:
R> # via summary statistics for the PS itself,
R> # via ND for the PS, and via ND for the log-odds ratio
R> ol1m<-mean(PSvec)
R> ol1sd<-sd(PSvec)
R> ol1min<-min(PSvec)
R> ol1max<-max(PSvec)
R> #
R> PS1<-PSvec[1:n1]
R> PS0<-PSvec[(n1+1):n]
R> my1<-mean(PS1)
R> my0<-mean(PS0)
R> sy1<-sd(PS1)
R> sy0<-sd(PS0)
R> ol2<-(my1-my0)/sqrt(sy1^2 + sy0^2)
R> #
R> lOddsvec<-predict(logit.out,newdata=NULL) #produces X*beta, the log-odds
R> #
R> L1<-lOddsvec[1:n1]
R> L0<-lOddsvec[(n1+1):n]
R> my1<-mean(L1)
R> my0<-mean(L0)
R> sy1<-sd(L1)
R> sy0<-sd(L0)
R> ol3<-(my1-my0)/sqrt(sy1^2 + sy0^2)
R> tt<-data.frame(col1=c("fitted prop. score", "ND for prop. score","ND for log odds"),
                 col2=c(mean(PSvec),ol2,ol3),
                 col3=c(round(sd(PSvec),digits=3),"-","-"),
                 col4=c(round(min(PSvec),digits=3),"-","-"),
                 col5=c(round(max(PSvec),digits=3),"-","-"))
R> colnames(tt)<-c("estimator","mean","std","min","max")
```

Table 1. summary stats for fitted propensity scores

estimator	mean	std	min	max
fitted prop. score	0.416	0.068	0.204	0.677
ND for prop. score	0.198	-	-	-
ND for log odds	0.199	-	-	-

All fitted propensity scores are well within the "comfort range" of 0.1-0.9. In addition, both ND scores fall below the "threshold for trouble" of 0.25, which is also reassuring. We can now compute the ATE and ATT estimates using the *inverse-weighting* expressions from our lecture notes.

3. Estimation via Propensity Score Weighting

```
R> px<-PSvec
R> w<-as.matrix(train)
R> y<-as.matrix(re78)
R> #make sure to re-define y compared to the PS model above!
R> int<-px*(1-px)
R> mp<-mean(((w-px)*y)/int)
R> #
R> rho<-n1/n
R> int<-rho*(1-px)
R> mpATT<-mean(((w-px)*y)/int)
```

As for the regression adjustment case, the trickiest part is getting consistent standard errors for these estimates. I prefer again the bootstrap approach, as it accounts for all layers of uncertainty: in the PS step, in the estimation step, and with respect to the distribution of **x** in the population.

```
R> R<-1000 #number of bootstrap replications
R> out<-rep(0,R) #will collect ATE result for each replication
R> outATT<-rep(0,R) #will collect ATT result for each replication
R> X<-cbind(age,educ,black,hisp,married,re74,re75)
R> com<-cbind(w, y, X)
R> for (i in 1:R) {
   int<-com[sample(nrow(com),n,replace=TRUE), ]
   #sample n id's with replacement (this allows for multiple entries)
   wr<-int[,1]
   yr<-int[,2]
   Xr<-data.frame(int[,3:dim(com)[2]])
   #note: Xr will automatically inherit all variable names from X
   # run logit, get fitted PS scores
   logit.out<-glm(wr~Xr$age+Xr$educ+Xr$black+Xr$hisp+Xr$married+Xr$re74+Xr$re75,
                  data=Xr,family=binomial)
   #this is much faster than attaching the entire data frame at each iteration
   PSr<-predict(logit.out,newdata=NULL,type="response")
   #
   int<-PSr*(1-PSr)
   out[i]<-mean(((wr-PSr)*yr)/int)
   #
   int<-rho*(1-PSr)
   outATT[i]<-mean(((wr-PSr)*yr)/int)
 }
R> semp<-sd(out)
R> tmp<-mp/semp
R> #
R> sempATT<-sd(outATT)
R> tmpATT<-mpATT/sempATT
```

4. Estimation via Regression on the Propensity Score

For comparison purpose, let's also derive these estimates using regression on the propensity score, first with separate regressions for controls and treated, then with a pooled regression.

```
R> y<-as.matrix(re78)
R> X<-cbind(rep(1,n),PSvec)
R> #
R> y1<-as.matrix(y[1:n1])
R> y0<-as.matrix(y[(n1+1):n])
R> X1<-X[1:n1,]
R> X0<-X[(n1+1):n,]
R> #
R> b1<-solve((t(X1)) %*% X1) %*% (t(X1) %*% y1)
R> b0<-solve((t(X0)) %*% X0) %*% (t(X0) %*% y0)
R> #
R> #ATE
R> y1p<-X%*%b1 #create predictions for treated outcome for ALL observations
R> y0p<-X%*%b0 #create predictions for UNtreated outcome for ALL observations
R> mpr<-mean(y1p-y0p)
R> #ATT
R> mprATT<-mean(y1p[1:n1]-y0p[1:n1])
R> #
R> #
R> #run bootstrap to get s.e.'s
R> ################################
R> R<-1000 #number of bootstrap replications
R> out<-rep(0,R) #will collect ATE result for each replication
R> outATT<-rep(0,R) #will collect ATT result for each replication
R> Xfull<-cbind(age,educ,black,hisp,married,re74,re75)
R> com<-cbind(w, y, Xfull)
R> #
R> #
R> for (i in 1:R) {
   # logit part
   ##################
   int<-com[sample(nrow(com),n,replace=TRUE), ]
   #sample n id's with replacement (this allows for multiple entries)
   wr<-int[,1]
   yr<-int[,2]
   Xr<-data.frame(int[,3:dim(com)[2]])
   #note: Xr will automatically inherit all variable names from X
   # run logit, get fitted PS scores
   logit.out<-glm(wr~Xr$age+Xr$educ+Xr$black+Xr$hisp+Xr$married+Xr$re74+Xr$re75,
                  data=Xr,family=binomial)
   #this is much faster than attaching the entire data frame at each iteration
   PSr<-predict(logit.out,newdata=NULL,type="response")
   #
   #regression part
   ###################
   Xr<-cbind(rep(1,n),PSr)
   y1r<-yr[1:n1]
   X1r<-Xr[1:n1,]
   b1r<-solve((t(X1r)) %*% X1r) %*% (t(X1r) %*% y1r)
   #
   y0r<-yr[(n1+1):n]
   X0r<-Xr[(n1+1):n,]
   b0r<-solve((t(X0r)) %*% X0r) %*% (t(X0r) %*% y0r)
   #
   y1rp<-Xr%*%b1r
   y0rp<-Xr%*%b0r
   #ATE
   out[i]<-mean(y1rp-y0rp)
   #ATT
   outATT[i]<-mean(y1rp[1:n1]-y0rp[1:n1])
 }
R> sempr<-sd(out)
R> tmpr<-mpr/sempr
R> #
R> semprATT<-sd(outATT)
R> tmprATT<-mprATT/semprATT


R> #pooled regression
R> ##################
R> X<-cbind(rep(1,n),w,PSvec)
R> bols<-solve((t(X)) %*% X) %*% (t(X) %*% y)
R> e<-as.vector(y-X%*%bols)
R> S<-diag(e^2)
R> Vb<-solve((t(X))%*%X) %*% t(X) %*% S %*% X %*% solve((t(X))%*%X)
R> se=sqrt(diag(Vb))
R> tval=bols/se
R> #
R> mprP<-as.vector(bols[2])
R> semprP<-as.vector(se[2])
```

```
R> tmprP<-as.vector(tval[2])
R> #
R> mprATTP<-mprP
R> semprATTP<-semprP
R> tmprATTP<-tmprP
```

Table 2. Estimation results

estimator	estimate	s.e.	t-value
ATE via PSw	1.627	0.626	2.601
ATE via sep. PSreg	1.620	0.636	2.547
ATE via pooled PSreg	1.682	0.661	2.545
ATT via PSw	1.798	0.660	2.724
ATT via sep. PSreg	1.823	0.635	2.871
ATT via pooled PSreg	1.682	0.661	2.545

```
R> proc.time()-tic
   user  system elapsed
   7.84    0.22    8.07
```

References

Wooldridge, J. (2010). *Econometric Analysis of Cross Section and Panel Data*, MIT Press.