PLSC 30600 - Lab 4 - Inverse Probability Treatment Weighting (IPTW) and Bootstrapping

01/27/2023

Analyzing an observational study: Keriakes et. al. (2000)

- This lab looks at an observational study looking at the effect of an experimental drug on survival rates after percutaneous coronary interventions (PCI) - interventions to open blocked coronary arteries (via angioplasty + insertion of a stent).
- Keriakes et. al. (2000) look at the effect of a drug called abciximab, a type of antiplatlet drug used to prevent blood clots during these coronary artery interventions.
- The dataset consists of 996 patients under observation at Ohio Heart Health, Christ Hospital, Cincinatti in 1997. Each patient received a percutaneous coronary intervention and was under observation for at least 6 months.
- At the end of the 6 month period, survival was recorded. Some patients received treatment with abciximab but in a non-random manner. Patients who doctors believed had more severe cases of heart disease were more likely to receive the drug.

```r
pci <- read_csv("pci.csv")
View(pci)
```

The relevant variables are

- `sixMonthSurvive` - Survival at 6 months - Main outcome of interest
- `abcix` - Treatment with abciximab - Main treatment of interest

The other observed pre-treatment covariates are

- `stent` - Coronary stent deployment; numeric, with 1 meaning YES and 0 meaning NO.
- `height` - Height in centimeters; numeric integer from 108 to 196.
- `female` - Female gender; numeric, with 1 meaning YES and 0 meaning NO.
- `diabetic` - Diabetes mellitus diagnosis; numeric, with 1 meaning YES and 0 meaning NO
- `acutemi` - Acute myocardial infarction within the previous 7 days; numeric, with 1 meaning YES and 0 meaning NO.
- `ejecfrac` - Left ejection fraction; numeric value from 0 percent to 90 percent.
- `ves1proc` - Number of vessels involved in the patient's initial PCI procedure; numeric integer from 0 to 5.

Balance checks before adjustment

Our naive estimate of the treatment effect without adjustment suggests a small positive effect on survival probability (about 3 pp).

```r
lm_robust(sixMonthSurvive ~ abcix, data=pci)
```

```
##               Estimate Std. Error   t value  Pr(>|t|)   CI Lower   CI Upper  DF
## (Intercept) 0.94966443 0.01268657 74.855861 0.0000000 0.92476889 0.97455997 994
## abcix       0.03457626 0.01353525  2.554534 0.0107811 0.00801531 0.06113721 994
```

- However, we have reason to believe this is biased for the true ATE of abciximab treatment since it was administered to patients with more severe illness. Let's diagnose the covariate balance.
- The R package `cobalt` is really useful for generating balance tables for weighting estimators - it does a lot of the annoying pre-processing automatically (like standardizing the variables to have a standard deivation of 1).

```r
# Load in cobalt
library(cobalt)
```

```
## Warning: package 'cobalt' was built under R version 4.2.2
```

```
##  cobalt (Version 4.4.1, Build Date: 2022-11-03)
```

```r
# Subset the covariates we want to a data-frame
pci_covs <- pci %>% select(stent, female, diabetic, height, acutemi, ejecfrac, ves1proc)
View(pci_covs)

# cobalt::bal.tab() will take a matrix of covariates and a treatment indicator and give standardized me
# it doesn't do t-tests by design (since they can be misleading and are discouraged by more recent work
# standardized mean differences above .1 are often a good threshold to be concerned (https://pubmed.ncb
balance_tab <- bal.tab(pci_covs, treat=pci$abcix, s.d.denom="pooled")
balance_tab
```

```
## Balance Measures
##             Type Diff.Un
## stent     Binary  0.1210
## female    Binary -0.0550
## diabetic  Binary -0.0636
## height   Contin. -0.0003
## acutemi   Binary  0.1187
## ejecfrac Contin. -0.1821
## ves1proc Contin.  0.4273
## 
## Sample sizes
##     Control Treated
## All     298     698
```

```r
# We can look not only at the differences in means but in differences of higher-order moments
balance_tab_sqd  <- bal.tab(pci_covs, treat=pci$abcix, s.d.denom="pooled", poly=2)
balance_tab_sqd
```

```
## Balance Measures
##              Type Diff.Un
## stent      Binary  0.1210
## female     Binary -0.0550
## diabetic   Binary -0.0636
## height    Contin. -0.0003
## acutemi    Binary  0.1187
## ejecfrac  Contin. -0.1821
## ves1proc  Contin.  0.4273
## height²   Contin.  0.0003
## ejecfrac² Contin. -0.2022
## ves1proc² Contin.  0.3974
##
## Sample sizes
##     Control Treated
## All     298     698
```

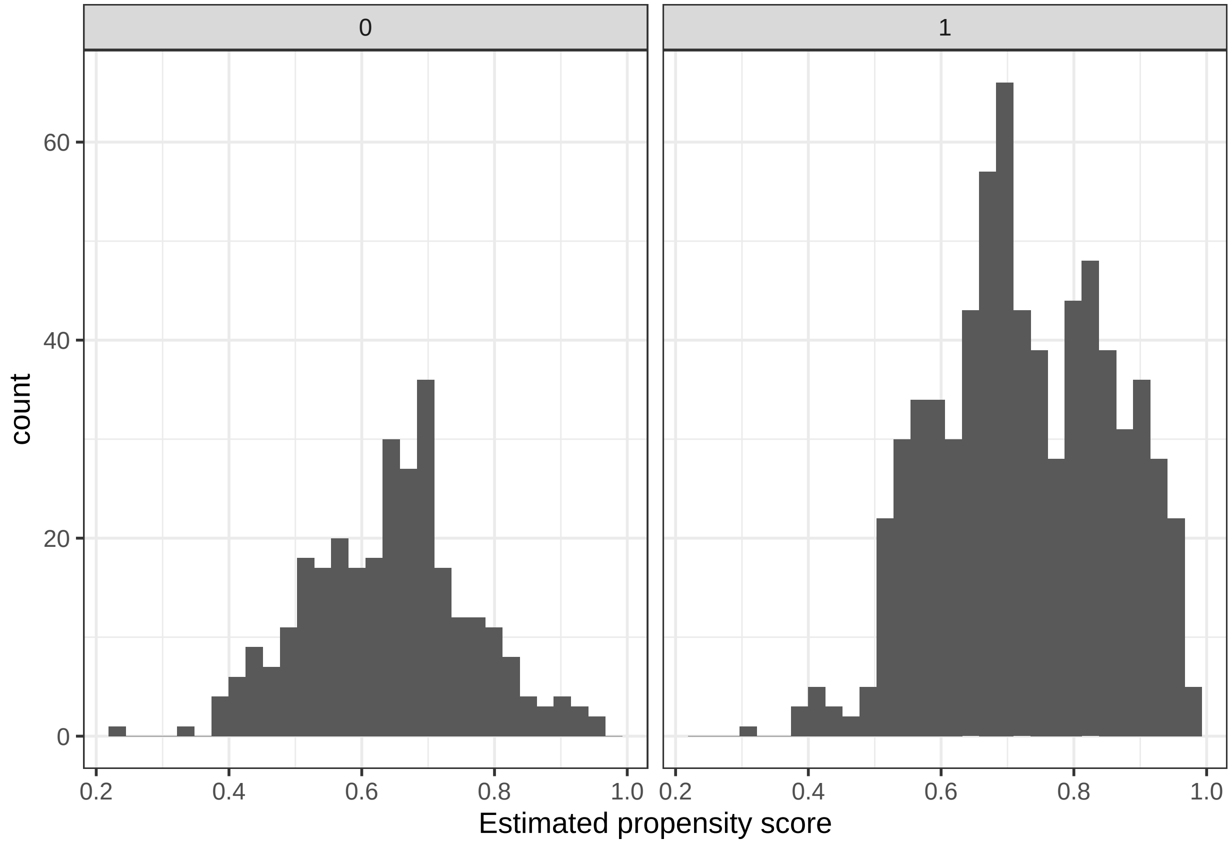
Propensity score estimation

Now let's estimate the propensity score model. Let's try the easiest model with no interactions or polynomial terms in the linear predictor

```
# Fit the logistic regression model
pscore_model <- glm(abcix ~ stent + female + diabetic + height + acutemi + ejecfrac + ves1proc,
                    data=pci,
                    family=binomial(link="logit"))

# Get the propensity scores for each observation
pci$e <- predict(pscore_model, type = "response") # type = "response" gives us the probabilities
View(pci)

# Let's see the histogram of the propensity scores among treated and control
pci %>% ggplot(aes(x=e)) +
  geom_histogram(bins=30) +
  facet_wrap(~abcix) +
  xlab("Estimated propensity score") +
  theme_bw()
```



This is really quite good in terms of overlap (if we got the model right). Not a lot of 0 or 1 propensity scores. Now let's make the Inverse Probability Treatment Weighting (IPTW) weights

```
# Difference in covariates
pci %>%
  group_by(abcix) %>%
  summarise(stent = mean(stent),
            height = mean(height),
            female = mean(female),
            diabetic = mean(diabetic),
            acutemi = mean(acutemi),
            ejecfrac = mean(ejecfrac),
            ves1proc = mean(ves1proc))
```

```
## # A tibble: 2 x 8
##   abcix stent height female diabetic acutemi ejecfrac ves1proc
##   <dbl> <dbl>  <dbl>  <dbl>    <dbl>   <dbl>    <dbl>    <dbl>
## 1     0 0.584   171.  0.386    0.268  0.0604     52.3     1.20
## 2     1 0.705   171.  0.331    0.205  0.179      50.4     1.46
```

```
# Generate the weights
pci$wt <- NA
pci$wt[pci$abcix == 1] <- mean(pci$abcix==1)/pci$e[pci$abcix==1]
pci$wt[pci$abcix == 0] <- mean(pci$abcix==0)/(1 - pci$e[pci$abcix==0])
View(pci)
```

```
# Generate a point estimate
iptw_est <- lm_robust(sixMonthSurvive ~ abcix, data=pci, weights=wt)
point_wtd <- coef(iptw_est)[2]
point_wtd
```

```
##      abcix
## 0.06609811
```

Recall that our unadjusted estimate was around 3 percentage points. After adjustment, the estimated effect is closer to 6 percentage points. This is in line with our expectations that the selection-into-treatment bias negatively biased the naive difference-in-means since patients with worse conditions were more likely to get treatment.

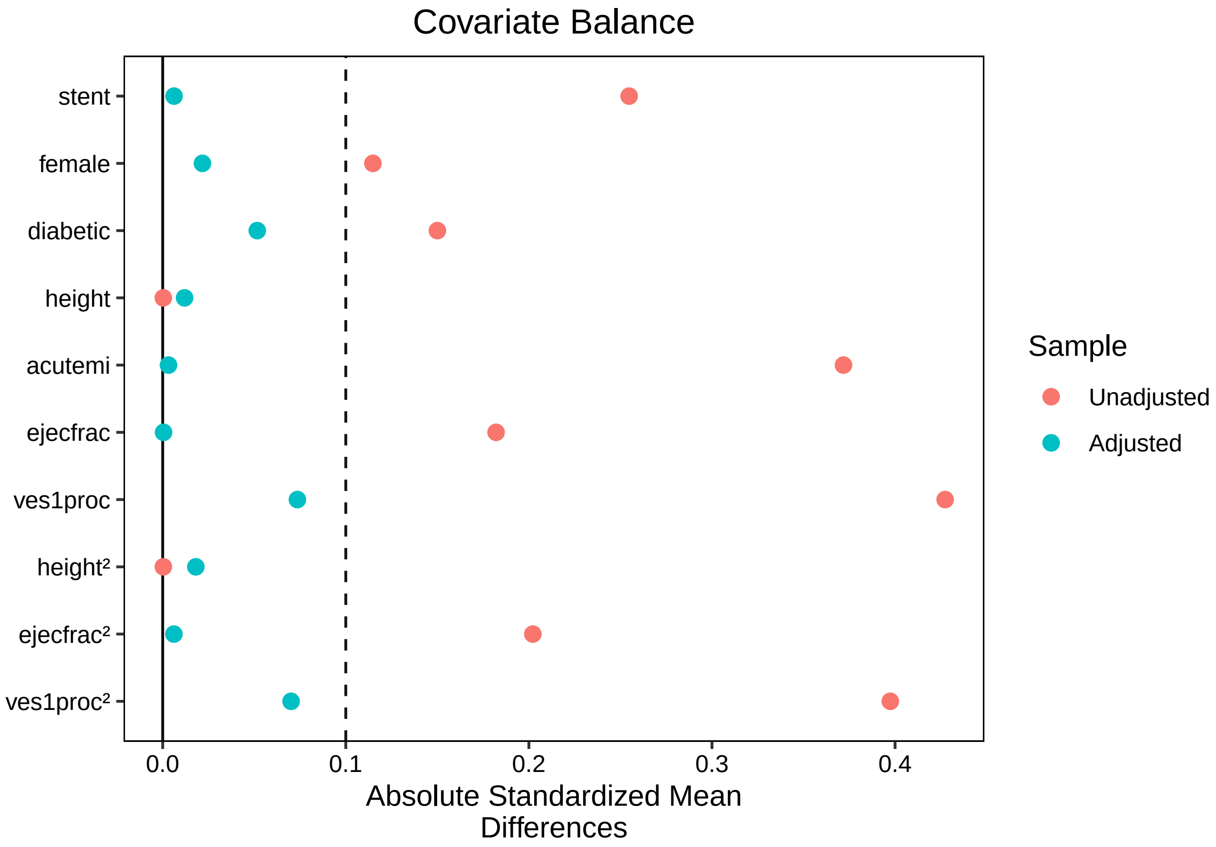
Balance checks after adjustment

We'll come back to inference in a second. But first, let's see how well covariate balance has improved with the weights.

```
# Pass the IPTW weights
balance_tab_wt <- bal.tab(pci_covs, treat=pci$abcix, s.d.denom="pooled", weights=pci$wt)
balance_tab_wt
```

```
## Balance Measures
##             Type Diff.Adj
## stent     Binary   0.0029
## female    Binary   0.0104
## diabetic  Binary  -0.0219
## height   Contin.  -0.0120
## acutemi   Binary  -0.0010
## ejecfrac Contin.  -0.0005
## ves1proc Contin.  -0.0736
##
## Effective sample sizes
##            Control Treated
## Unadjusted  298.    698.
## Adjusted    199.68  671.09
```

```
# A mega useful visualization tool to compare unadjusted vs. adjusted
# is the "love" plot (named after biostatistician Thomas Love)
iptw_love_plot <- love.plot(pci_covs, treat=pci$abcix, s.d.denom="pooled", abs=T, poly=2,
                            binary = "std", weights=pci$wt, thresholds= c(m=.1))
iptw_love_plot
```



Note that we even get improvement on the higher-order moments of some of the continuous covariates even though we didn't include them in the model. However, we don't get improvement *everywhere* - post-weighting balance is slightly worse for height (on which there was basically no imbalance to begin with). But overall, weighting does a pretty good job of attenuating the difference in the *observed* covariates between treated and control. Of course, you should keep in mind that this is only addressing balance on covariates that we *observe* - if there's unobserved confounding (which we're assuming there's not), we could be making it worse!

Bootstrapping

Recall again that bootstrapping is a way of approximating the sampling distribution of an estimator and estimating features of it (such as the variance), by resampling from our sample. With independent observations, the nonparametric bootstrap repeatedly resamples observations *with replacement* from the sample and computes an estimate for each resample.

```r
set.seed(60637)
nBoot <- 1000 # Number of iterations
ate_boot <- rep(NA, nBoot) # Placeholder to store estimates

# For each iteration
for(boot in 1:nBoot){

  # Resample rows with replacement
  pci_boot <- pci[sample(1:nrow(pci), nrow(pci), replace=T),] #replace = T is key!

  # Fit the propensity score model on the bootstrapped data
  pscore_model_boot <- glm(abcix ~ stent +  female + diabetic + height +
                             acutemi + ejecfrac + ves1proc, data=pci_boot,
                           family=binomial(link="logit"))

  # Save the propensities
  pci_boot$e_boot <- predict(pscore_model_boot, type = "response")

  # Calculate the weights
  pci_boot$wt_boot <- NA
  pci_boot$wt_boot[pci_boot$abcix == 1] <- mean(pci_boot$abcix==1)/pci_boot$e[pci_boot$abcix==1]
  pci_boot$wt_boot[pci_boot$abcix == 0] <- mean(pci_boot$abcix==0)/(1 - pci_boot$e_boot[pci_boot$abcix==

  # weighted difference-in-means
  boot_reg <- lm_robust(sixMonthSurvive ~ abcix, data=pci_boot, weights=wt_boot)

  # Store the weighted difference-in-means
  ate_boot[boot] <- coef(boot_reg)[2]

}

# Take the SD of the ate_boot to get our estimated SE - can do asymptotic inference
sd(ate_boot)
```

```
## [1] 0.02826734
```

```
# Asymptotic 95\% CI
c(point_wtd - qnorm(.975)*sd(ate_boot),
  point_wtd + qnorm(.975)*sd(ate_boot))
```

```
##      abcix      abcix
## 0.01069514 0.12150109
```

```
# Can also take quantiles to get CIs directly from the bootstrapped distribution (esp. if skewed)
quantile(ate_boot, c(.025, .975))
```

```
##       2.5%      97.5%
## 0.01395915 0.12283017
```

Our 95% confidence interval does not include 0 – we'd reject the null of no ATE at $\alpha = .05$.

Challenge problem

Play around with different specifications for the propensity score model (including interactions or higher-order polynomial terms - for polynomials use `I()` in the formula around the expression to make sure R processes it correctly). Compare the change in balance relative to the model without interactions or polynomials - is there a specification where we can do *even better* on balance?