ECON 293 - Problem Set 1

Jacob Light

4/29/2020

Part 1 - Randomized Experiment

```
# Start with STAR dataset, constructed using STAR_students.sav and Comparison_students.sav
# datasets. Restrict to observations for which first grade test score are available, select
# score data, student id, and common demographic characteristics (race, gender, birth year).
# Assume, for this exercise, that this sample represents a randomized experiment.
ate <- function(df) {
  df %>%
    group_by(treatment, test) %>%
    mutate(score = if_else(treatment == 0, -score, score)) %>%
    summarize(avg = mean(score), var = var(score), count = n()) %>%
    mutate(se = sqrt(var / (count - 1))) %>%
    group_by(test) %>%
    summarize(ate = sum(avg),
              se = sum(se),
              ci_low = ate - 1.96 * se,
              ci_high = ate + 1.96 * se)
}
summary <- ate(star_data)

kable(summary,
    caption = 'Baseline ATE - STAR Sample',
    digits = 2)
```

1. For my replication and modification study, I estimate the effect of assignment to a small class on first-grade reading and math scores using data from the the Tennessee STAR experiment . For the purposes of this exercise, I assume that treatment selection and attrition are entirely random and proceed with a sample comprised of students for whom I can observe treatment status, first-grade reading and math scores, age, sex, and birth year. Table 1 summarizes the baseline ATE estimate, standard error, and confidence intervals for first grade reading and math proficiency scores. The scores measure the percent of grade level standards a student mastered. The treatment effect can be interepreted as suggesting that assignment to a smaller classroom in the STAR experiment changed expected reading proficiency by and math proficiency by .

Table 1: Baseline ATE - STAR Sample

test	ate	se	ci_low	ci_high
math	3.93	0.86	2.25	5.61
read	6.93	1.03	4.91	8.94

Table 2: Difference in Means Test - P(Free Lunch) by Treatment Group

avg_treat	avg_control	dif	se	ci_low	ci_high
0.4755	0.5065	-0.0303	0.0232	-0.0759	0.0152

Table 3: Difference in Performance by Free Lunch Status

free_lunch	test	ate	se	ci_low	ci_high
0	math	2.6	0.9	1	4
1	math	4.8	1.4	2	8
0	read	4.4	1.2	2	7
1	read	8.8	1.6	6	12

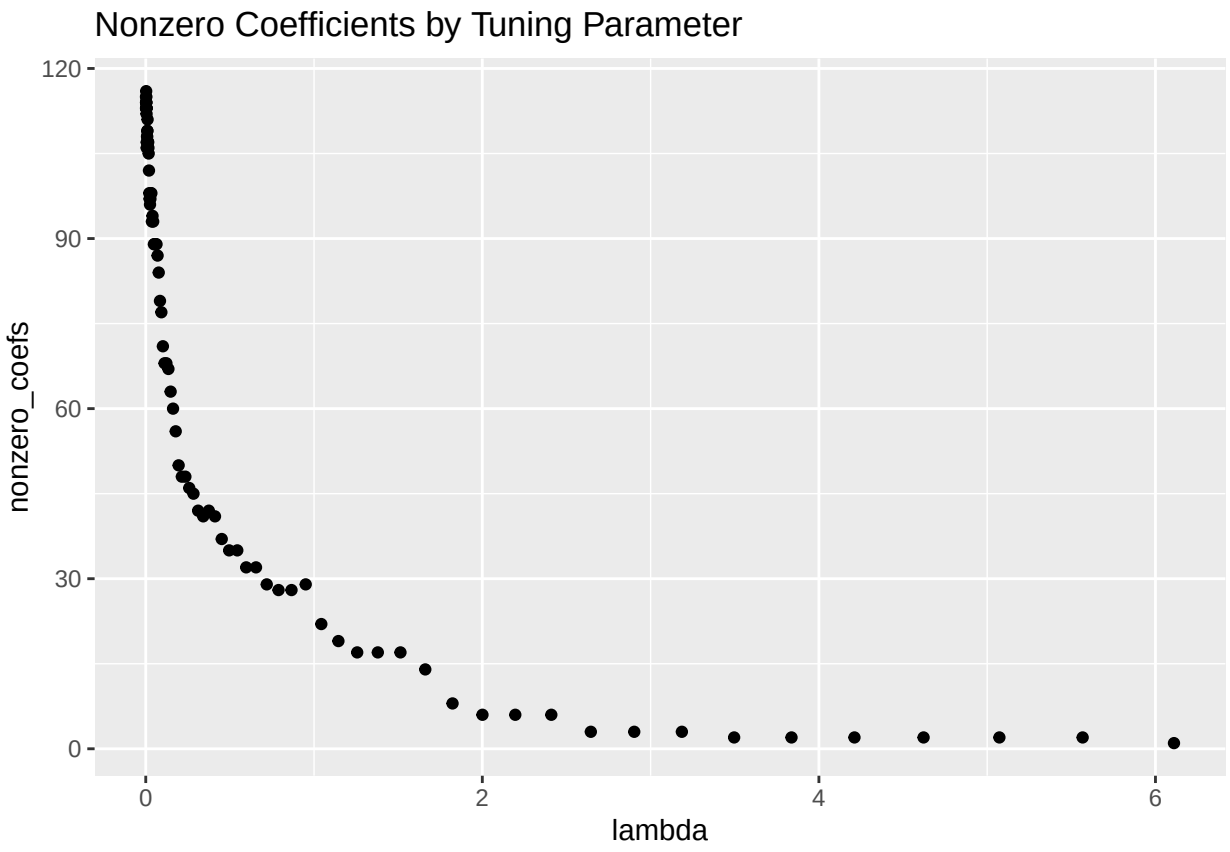
2. Under randomization, 48% of treated students and 51% of control students are eligible for free/reduced price lunch. This difference is not significant Suppose that the study did not properly balance socioeconomic status in the sample, such that students in the treatment group were substantially more likely to be free/reduced price lunch students than control students. Following the procedure in the tutorial, I drop 50% of free/reduced price students from the control group and 50% of non-free/reduced price students from the treatment group, such that the respective groups are more heavily skewed on the free/reduced price dimension. From the remaining observations, I further confound on gender: I drop 20% of boys in the control group and 20% of girls in the treatment. The table below summarizes the ATE in reading and math scores for this new confounded sample. The counfounded treatment effect estimates are substantially higher for both reading and math scores, although overlap on covariates has been preserved. The table below estimates confounded treatment effects.

3. The table below summarizes treatment effect estimates using inverse propensity weighting, OLS regression, and double robust analysis. The STAR data are somewhat limited for estimating causal questions using a large set of covariates - the STAR experiment staggered treatment start (some students drop out of the kindergarten cohort and are not observed again, some students enter the study during first grade), so there is only a small set of baseline demographic controls not affected by treatment that I can use to generate interactions. For the purposes of this exercise, I expand the number of covariates by interacting all of the baseline covariates (race, free/reduced lunch status, gender, birth year, and a vector of teacher characteristics), then re-run the three estimation procedures. With only slight modifications to the regression procedures, the effect on estimated ATE is relatively small. The initial IPW and expanded dataset IPW estimates are anomalous. The table also includes lasso and regression tree model estimates. After introducing a large set of interaction terms in the previous regressions, we might be concerned that we are over-fitting our model to the data. While a highly-fitted model increases in-sample fit, we also want the model to be flexible for out-of-sample fit. Thus, in lasso regression, we introduce a penalty for over-fitting to force the model to choose only the most informative covariates to include in the analysis. The lasso penalty function sends any unimportant parameters (specifically, if the benefit from fit is less than the size of the parameter estimate) to 0. The size of this penalty is not obvious from theory, so we would analytically choose the penalty term λ by splitting the sample into test and train samples and choosing the term $\hat{\lambda}$ that maximizes the out-of-sample fit on the test sample. In the figure below, I plot λ against the number of nonzezro terms included in the regression

Table 4: ATE Confounded on Free/Reduced Lunch Status

test	ate	se	ci_low	ci_high
math	4.31	0.92	2.51	6.11
read	6.99	1.11	4.83	9.16

- as λ grows, the penalty for including a relatively uninformative covariate shrinks and we begin to push those coefficient estimates to 0. Next, I run a regression tree model on the math test score data (restrict to only the math scores for simplicity). I include all of the covariates (without interactions) in the model, as the regression tree allows for the nonlinearities I simulated with interaction terms. At each branch of the regression tree, the algorithm splits a region of the sample into two sub-regions and calculates the average math score in the region. The algorithm can run until each observation is its own leaf, which maximizes in-sample fit but raises concerns about over-fitting. Thus, I split the data into train and test samples and use cross-validation to find the number of splits that maximizes out-of-sample fit. I estimate the regression tree ATE using this tuning parameter. I plot the first few branches of the regression tree below.



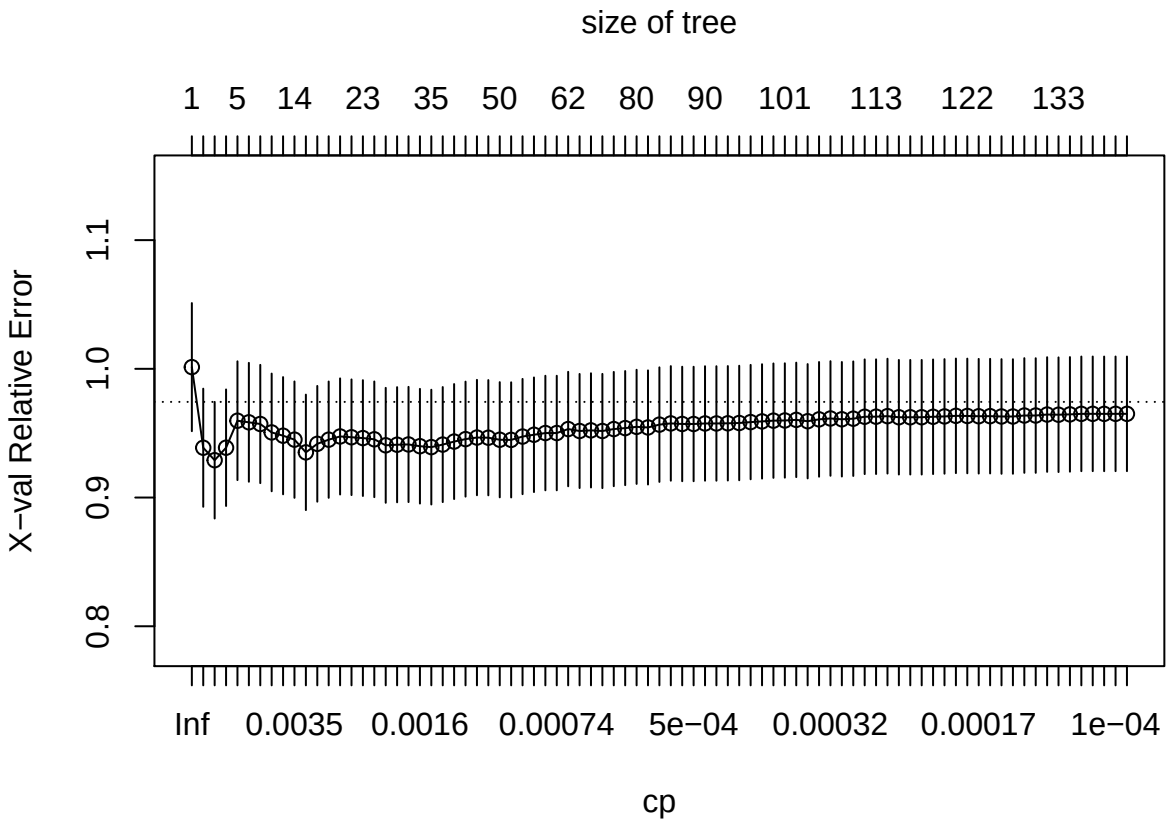
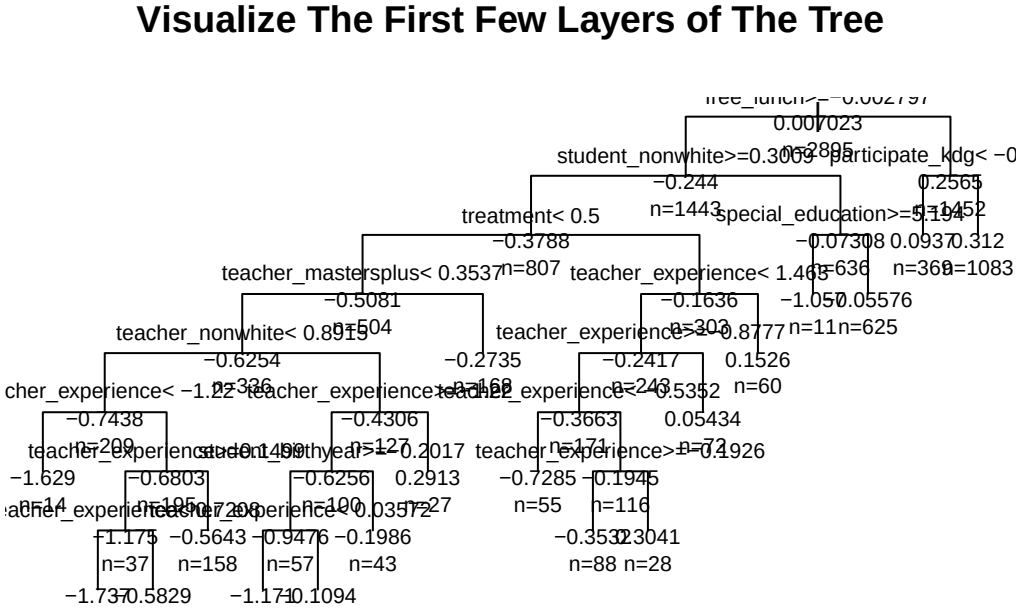

size of tree
1 5 14 23 35 50 62 80 90 101 113 122 133
X−val Relative Error
0.8 0.9 1.0 1.1
Inf 0.0035 0.0016 0.00074 5e−04 0.00032 0.00017 1e−04
cp

Table 5: Treatment Effect Estiamtes - Math Score

type	ATE	lower_ci	upper_ci
OLS	2.35	1.08	3.63
OLS w/ Interactions	2.35	1.04	3.66
Inverse Propensity Weight	1.09	-5.56	7.73
Inverse Propensity Weight w/ Interactions	0.82	-5.86	7.50
Double Robust	-15.39	-98.47	67.68
Double Robust w/ Interactions	31.64	-212.25	275.53
Lasso OLS	3.81	NA	NA
Lasso IPW	-34.22	-133.21	64.77
Regression Tree	0.06	NA	NA



4. The previous tables compare treatment effect estimates for the confounded dataset using the methods specified in the problem set (OLS, inverse propensity weight, double robust, lasso, regression tree, and higher-dimension models). Clearly, most of the answers are not reasonable or what I would have expected, so I'll discuss some of the issues I encoutered in estimation in the hopes of uncovering some areas where I made mistakes. The esimates that produce the most reasonable treatment effect estimates are OLS and the Lasso OLS. We would expect OLS to better fit the underlying data because OLS does not include any regularization as we increase the dimensions of the covariate matrix. The inverse propensity weight regressions are not working as we might expect. I am not sure if this is a result of the trimming/confounding exercise or some underlying bug in my code. The regression tree estimates for the math score treatment effect are also quite small - I do not have a strong prior to explain this issue.

Table 6: Treatment Effect Estiamtes - Reading Score

type	ATE	lower_ci	upper_ci
OLS	4.13	2.60	5.66
OLS w/ Interactions	4.08	2.52	5.64
Inverse Propensity Weight	1.00	-5.20	7.21
Inverse Propensity Weight w/ Interactions	0.73	-5.50	6.96
Double Robust	-41.83	-329.74	246.08
Double Robust w/ Interactions	24.91	-423.05	472.86
Lasso OLS	4.59	NA	NA
Lasso IPW	-27.40	-130.37	75.56

Part 2 - Propensity Stratification

1. Estimating propensity scores extracts from a treatment assignment indicator any portion of assignment that can be explained by non-random factors. Among observations with identical propensity scores, treatment status can be considered random and we can apply our usual potential outcomes framework to estimate treatment effects local to individuals with a given propensity score. Stratifying a sample on propensity scores subsets a larger sample into subsamples with similar propensity scores, where in sufficiently large samples with sufficiently many strata, we approximate potential outcomes setting of a random sample. We'd then take a weighted average of consistently-estimated point estimates, which will yield a consistent estimate of the ATE.

```
ate_estimate <- function(X, prop, K = 5) {
  # x is a list object containing a matrix
  # of covariates, a vector of treatment dummies,
  # and a vector of outcomes. K is the number
  # of strata chosen by the user

  # 1 - True ATE
  tib <- tibble(y = X$outcome, w = X$treatment, e_x = X$propensity) %>%
    mutate(invprop = y * w / e_x - y * (1 - w) / (1 - e_x))
  ate_true <- tibble(type = "True ATE",
                         point_estimate = mean(tib$invprop))
                         # sq_error = (mean(tib$invprop) - ate_meandif$point_estimate) ^ 2

  # 2 - Inverse propensity weight
  tib <- tibble(y = X$outcome, w = X$treatment, e_x = prop) %>%
    mutate(invprop = y * w / e_x - y * (1 - w) / (1 - e_x))
  ate_invprop <- tibble(type = "Inverse Propensity Weight",
                         point_estimate = mean(tib$invprop))
                         # sq_error = (mean(tib$invprop) - ate_meandif$point_estimate) ^ 2
                         # ci_low = mean(tib$invprop) - 1.96 *
                         #   sqrt(var(tib$invprop) / (dim(tib)[1] - 1)),
                         # ci_high = mean(tib$invprop) + 1.96 *
                         #   sqrt(var(tib$invprop) / (dim(tib)[1] - 1)))

  # 3 - Stratified treatment effect
    # a - stratify sample into K strata
    tib$quantile <- ntile(x = tib$e_x, n = K)

    # b - calculate within-stratum ATE
    ate_strat <- tib %>%
```

```
      group_by(w, quantile) %>%
      summarize(e_y = mean(y), n_w = n(), v = var(y)) %>%
      ungroup() %>%
      # Ensure that each stratum is populated by treated and control individuals
      group_by(quantile) %>%
      mutate(check = (max(w) == 1 & min(w) == 0)) %>%
      filter(check == TRUE) %>%
      select(-check) %>%
      mutate(v = v / n_w) %>%
      # Calculate local treatment effect
      mutate(e_y = if_else(w == 0, -1 * e_y, e_y)) %>%
      group_by(quantile) %>%
      summarize(effect = sum(e_y), n = sum(n_w), v = sum(v)) %>%
      mutate(ate = effect * n / sum(n),
             v = sqrt(v)) %>%
      select(ate) %>%
      sum() %>%
      as.numeric()
    ate_strat <- tibble(type = "Stratified ATE",
                        point_estimate = ate_strat)
                        # sq_error = (ate_strat - ate_meandif$point_estimate) ^ 2)

  return(list(ate_true = ate_true,
              ate_invprop = ate_invprop,
              ate_strat = ate_strat))

}
```

2. The above function `ate_estimate` returns a matrix continaing point estimates and confidence intervals for randomized, inverse propensity-weighted, and stratified ATEs. The function accepts as an argument the number of strata `K`, which is limited to be less than $2N$, the smallest possible number of strata in which a local treatment effect could be estimated. The function as written drops strata that do not contain at least one treated and at least one control observation and calculates the stratified ATE across all strata for which an ATE can be estimated. This approach is likely most appropriate in a context where treatment status is heavily confounded. If we are not going to control for the dissimilarity of treated and control observations, it might be more sensible to restrict the sample to a region where control and treated individuals are more comparable and think of the treatment effect as a local treatment effect. Alternatively, we might run into an issue of strata where the ATE cannot be estimated because the strata are too small and all observations have the same treatment status by chance. Dropping these strata should not introduce bias into the point estimate but will increase the variance.

```
# 1 - Generate dataset
set.seed(400)
datagen <- function(m = 3) {
  n <- 1000
  p <- 20
  X <- matrix(rnorm(n * p), n, p)
  propensity = pmax(0.2, pmin(0.8, 0.5 + X[,1]/3))
  W <- rbinom(n, 1, propensity)
  Y <- pmax(X[,1] + W * X[,2], 0) + rnorm(n)
  return(list(covariates = X,
              treatment = W,
```

Table 7: Compare Treatment Effect Estimates by Method

type	point_estimate
True ATE	0.3378496
Inverse Propensity Weight	0.0382073
Stratified ATE	0.1703122

```
                outcome = Y,
                propensity = propensity))
}
```

3. Using the code provided, I simulate a dataset. The average treatment effect is calculated in the table below.

```
data_estimator <- function(X) {
  # 1 - run logistic regression of W on X
  prop <- glm(X$treatment ~ X$covariates, family = 'binomial') %>%
    predict(type = 'response')

  # 2 - Estimate treatment effects
  ate_estimate(X, prop, 5)
}

# Generate data frame, apply data_estimator function, print confidence intervals
df <- datagen()
out <- data_estimator(df)
kable(out %>% bind_rows(),
      caption = 'Compare Treatment Effect Estimates by Method')
```

4. The table above estimates ATEs using inverse-propensity weighted and propensity-stratified methods. I estimate propensity scores using a logistic regression. I estimate the propensity-stratified ATE with 5 strata, as is common in the literature (e.g. Rosenbaum and Rubin 1983). The optimal choice of number of strata balances the bias-reduction of estimating local treatment effects and imprecision of strata estimates when strata are unbalanced or too small. An alternate approach might have been to use a machine learning method to balance the bias and variance reduction in the choice of the number of strata. The "True ATE" is an inverse-propensity weighted ATE using the propensity scores generated in the data generation code, rather than the logistic estimate.

```
# Repeat 20 times
sim_20 <- lapply(1:20, datagen) %>%
  lapply(data_estimator) %>%
  lapply(bind_rows) %>%
  bind_rows() %>%
  mutate(iter = cumsum(if_else(type == "True ATE", 1, 0))) %>%
  group_by(iter) %>%
  mutate(true = cumsum(if_else(type == "True ATE", point_estimate, 0)),
         sq_error = (point_estimate - true) ^ 2) %>%
  ungroup() %>%
  group_by(type) %>%
  summarize(mse = mean(sq_error))
```

Table 8: Mean Square Error by Method

type	mse
Inverse Propensity Weight	0.0189
Stratified ATE	0.0013
True ATE	0.0000

```
# Table
kable(sim_20,
      caption = "Mean Square Error by Method", digits = 4)
```

5. In the table above, I replicate the method in #4 on 20 simulated data frames and calcualte the MSE in average treatment effect. Stratification on propensity score minimizes the MSE relative to inverse propensity weighting - inverse propensity weighting introduces additional error by boosting certain observations and dampening others in the estimation. Thus, we'd expect the inverse propensity weighting method to be slightly noisier than the stratified method.