A quick exercise on adjustment

SICSS-UCLA 2020

Pablo Geraldo Bastías

June 18, 2020

Preliminaries

In what follows, I will demonstrate the use and compare different methods to estimate causal effects in observational data. Let's denote units $i, \dots, N$, with treatment $D \in \{0, 1\}$, and potential outcomes Y_d. In particular, as is common in this literature, we want to estimate an Average Treatment Effect on the Treated (ATT), defined as: $\mathbb{E}[\tau_i|D = 1] = \mathbb{E}[Y_{1i} - Y_{0i}|D = 1]$.

We also observe a serie of pre-treatment variables X_i. To estimate the ATT from observational data, in addition to SUTVA or consistency of the potential outcomes, we would need to assume the following:

- (Mean) Conditional independence of the control potential outcome and the treatment:

$$\mathbb{E}[Y_{0i}] \perp\!\!\!\perp D|X$$

- Positivity or common support

$$0 > P(D|X) > 1$$

Under these conditions, we can estimate conditional treatment effects $\mathbb{E}[Y_{1i} - Y_{0i}|X = x]$ for each X, and then recover the ATT using the adjustment formula:

$$\sum_{X_{D=1}} \mathbb{E}[Y|D, X]P[X_{D=1}]$$

Where $X_{D=1}$ represent the covariate distribution among the treated.

Lalonde data and experimental benchmark

We will use the classical data in Lalonde [1986]. This dataset is widely used to evaluate the performance of different methods to estimate causal effects using observational data (see, for example, Hainmueller [2012]), since it has the unique feature of providing an experimental component that could be used as a benchmark, and an observational component that can be used to test the proposed method.

The experimental portion of the data comes from the National Supported Work Demonstration (NSW), a labor training program that randomly selected participants to treatment ($N_t = 185$) and control ($N_c = 260$) conditions. The observational data, in addition to the treated group from the experiment, includes information about additional subjects from the Panel Survey of Income Dynamics (PSID), who are potential observational controls ($N_{c,obs} = 2490$).

The data includes 10 raw covariates: age, education (years), race (white, black, hispanic), marriage status (married/not), high school diploma (yes/no), and both previous earnings and unemployment (during 1974 and 1975). The outcome of interest is earnings in 1978. To better illustrate the difference between methods, and following Hainmueller [2012], I will use an expanded version of this covariates, including their squares and every two-way interaction (65 covariates in total).

```r
library("cobalt") # Package to assess covariate balance
library("causalsens") # lalonde experimental vs lalonde observational

####################
### DATA LOADING ###
####################

# First, let's load the experimental dataset (lalonde.exp)
data(lalonde.exp, package="causalsens")
# Then also the observational dataset (lalonde.psid)
data(lalonde.psid, package="causalsens")

########################
### BASELINE BALANCE ###
########################
library("ebal") # Entropy balance (and matrixmaker function)

# Let's expand the matrix of covariates
lalonde.exp.covs <- lalonde.exp %>% select(-"treat",-"re78") %>% matrixmaker()
lalonde.obs.covs <- lalonde.psid %>% select(-"treat",-"re78") %>% matrixmaker()
```

```
# Then list the covariates we will be using in the analysis
covs_raw <- lalonde.exp %>% select(-re78, -treat) %>% names()
covs <- names(lalonde.obs.covs)

# Create a single dataset with both obs and experimental data
# Just to plot balance in both
lalonde.mix <-
  rbind(lalonde.exp.covs %>%
          mutate(treat = lalonde.exp$treat, wexp=1, wobs=ifelse(treat==1,1,0)),
        lalonde.obs.covs[lalonde.psid$treat==0,] %>% # Observational controls
        mutate(treat=0, wexp=0, wobs=1)) # Weights for obs data
#names(lalonde.mix)

# Create "loveplot" to show covariate distribution in both samples
```

A first interesting inspection of the data is to assess the overall covariate balance in the different samples. As we can see in Figure 1, in the experimental sample all the covariates are very well balanced among treated and controls. In the observational sample, on the other hand, treated and control subjects are very different, in particular with respect to their previous earning trajectories.

```
# The bal.table function takes the raw data and weights
# And outputs a table of standardized difference between treated and controls
# Love.plot takes a bal.table object as inputs
# And transform it into a plot to facilitate presentation

bal.tab(f.build("treat", covs), data=lalonde.mix,
        estimand = "ATT", m.threshold =.1,
        disp.var.ratio=FALSE, disp.means=TRUE,
        weights = c("wexp","wobs")) %>% # Weight for experimental and observational sample
  # Note that if you comment out everything from the pipe and after
  # You will see the table in which the plot is based
  # And it contains more information than the plot
  love.plot(abs=FALSE,
            sample.names =
```

```r
              c("Combined","Experimental","Observational")) +
  labs(title= " ") + theme(legend.position = "top")
```

A second aspect that is worth noting are the estimated returns of the program, as we can see in Figure 2. The experimental benchmark is around 1,600 and 1,800 USD, depending if we use a simple difference in means or the Lin estimator[1]. The observational difference in means, as expected from the huge covariate imbalance, is completely off: -15,205 USD. The multiple regression results with the observational data are more interesting, however. Although the point estimate is way below the point estimate from the experimental benchmark (suggesting only a 4USD return), the confidence interval resulting from the regression model overlaps with the benchmark returns. That's it, the regression model do a good job in accounting for the uncertainty associated to estimate the effect of the program, providing a wide and uninformative confidence interval.

```r
#######################
### BASELINE MODELS ###
#######################
library("estimatr") # Lin estimator

# Bivariate regression on experimental data: 1,794 USD
exp_biv <- lm(re78 ~ treat, data=lalonde.exp)
# Lin estimator (treatment fully interacted) on experimental data: 1,583 USD
exp_lin <- lm_lin(f.build("re78", "treat"), covariates = f.build("", covs_raw), data=lalonde.exp)
# Bivariate linear regression on observational data: -15,205 USD
obs_biv <- lm(re78 ~ treat, data=lalonde.psid)
# Multiple regression with controls on observational data: 4.159 USD
obs_reg <- lm(f.build("re78", c("treat", covs_raw)), data=lalonde.psid)
# Combine results (Store in a matrix for plotting)
level_order <- c("Experimental (diff-in-means)","Experimental (Lin estimator)",
             "Observational (Regression)","Observational (diff-in-means)")

# Compile the resulting coefficients into a single matrix
coeffs <-
  tibble(estimator =
           factor((c("Experimental (diff-in-means)",
```

[1]This is basically a regression model with all raw covariates and their interactions with the treatment.

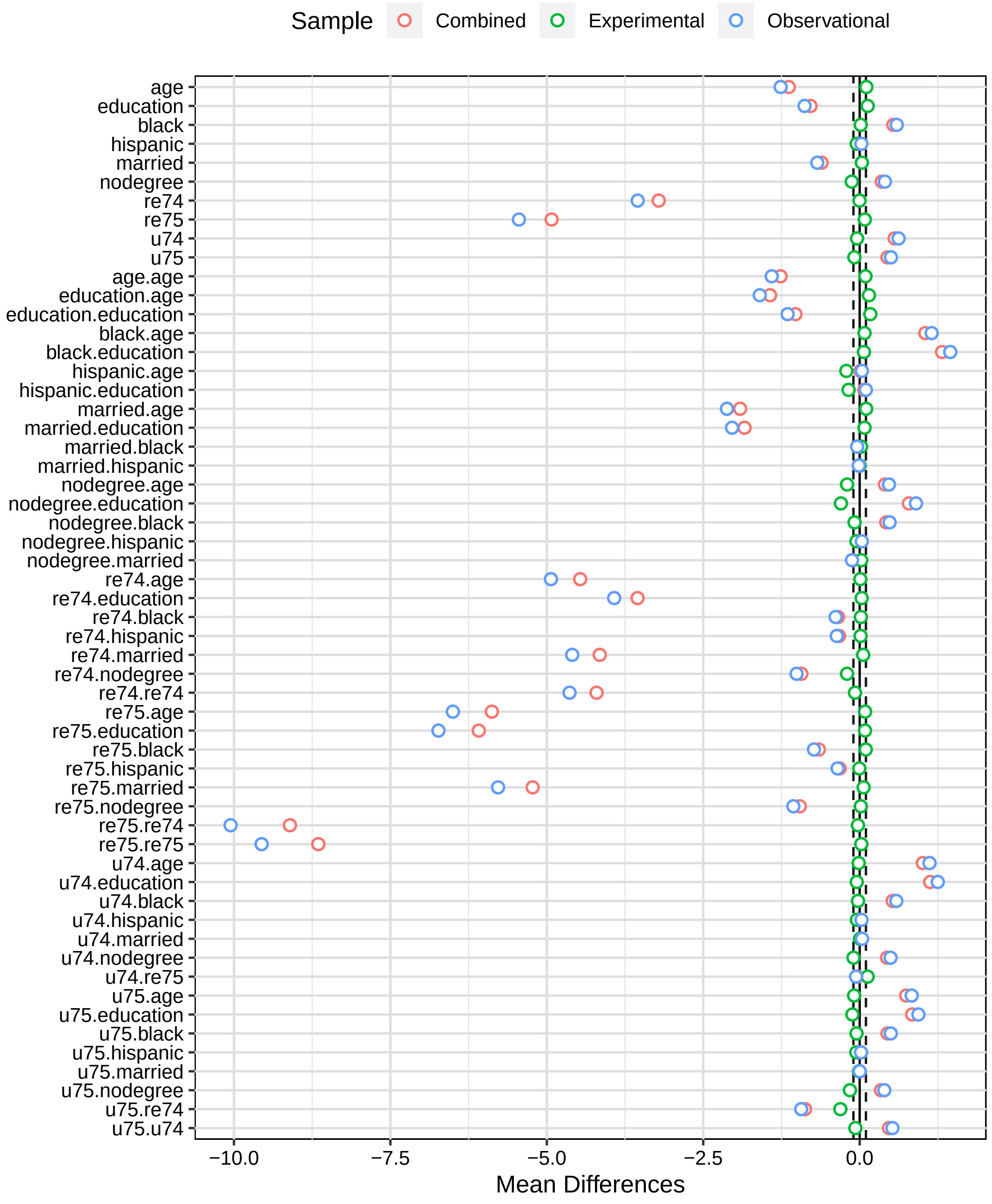

Figure 1: Covariate Balance by sample

```
            "Experimental (Lin estimator)",
            "Observational (Regression)",
            "Observational (diff-in-means)")), levels=level_order),
        coeff = c(exp_biv$coefficients[2],
                  exp_lin$coefficients[2],
                  obs_reg$coefficients[2],
                  obs_biv$coefficients[2]),
        se = c(summary(exp_biv)$coefficients[2,2],
               exp_lin$std.error[2],
               summary(obs_reg)$coefficients[2,2],
               summary(obs_biv)$coefficients[2,2]))

# Create the annotated plot
ggplot(coeffs, aes(x=coeff, y=estimator)) +
  geom_point() +
  # Error bars (Confidence intervals)
  geom_errorbarh(aes(x=coeff, y=estimator,
                     xmin=coeff-2*se, xmax=coeff+2*se),
               height=0) +
  geom_vline(xintercept = 0, linetype="dashed", color="red") +
  # Annotations (point estimates)
  annotate("text", x = -2500, y = 1, label = "1,794 USD") +
  annotate("text", x = -2500, y = 2, label = "1,583 USD") +
  annotate("text", x = -4000, y = 3, label = "4.159 USD") +
  annotate("text", x = -10000, y = 4, label = "-15,205 USD") +
  # Labels
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

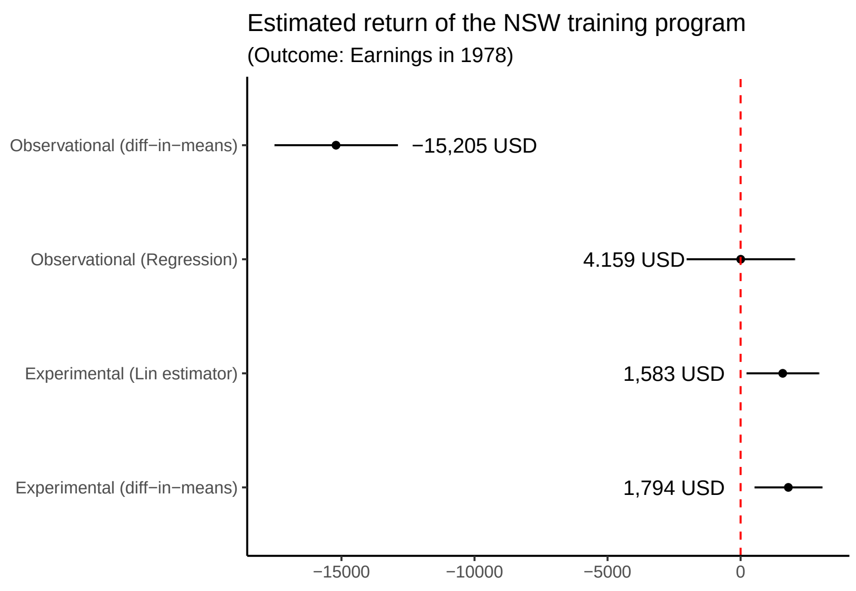

Figure 2: Estimated returns of the training program.

"Traditional" methods for estimating causal effects (i.e., matching and weighting).

In this section, I will demonstrate the use and performance of two “traditional” conditioning methods: inverse probability weighting (as implemented in the `weighIt` package by Greifer [2020]), and optimal full matching on the propensity score (using the `optmatch` package by Hansen et al. [2019]). I consider the main characteristic of “traditional” that they achieve balance *in expectation*, but not necessarily within sample.

I selected these two combinations for the following reasons. First, propensity score has played a key role in observational causal inference since the ’80s. Second, the more common variants when implementing propensity score analyses have been usually some form of either matching or weighting[2].

```
# First, add treatment and outcome variables to the expanded set of covariates
lalonde.obs <- cbind(lalonde.psid[,c("treat","re78")], lalonde.obs.covs)


library("WeightIt") # Weighit function
# Propensity Score Weighting (WeighIt package)
nsw_ipw <-
  weightit(f.build("treat", covs), # Formula: treatment ~ covariates
```

[2]Other uses are possible. For example, one can use the propensity score as a control variable in regression analysis, or use propensity score to stratify the sample.

```
           data = lalonde.obs, # Declare data
           estimand = "ATT", # Declare estimand
           method = "ps") # Weighting method
#summary(nsw_psw)

# Optimal Full matching (optmatch package)
library("optmatch")
# Create a propensity score using a logit with expanded covariates
ps <- glm(f.build("treat", covs), family = binomial, data = lalonde.obs)
# Create the optimal full matched sample
nsw_full <- fullmatch(ps, data = lalonde.obs)
#summary(nsw_full)
```

As we can see in Figure 3, both inverse probability weighting and optimal full matching greatly increase covariate balance with respect to the unadjusted observational comparison. Optimal full matching is slightly better in general, but the differences are not striking.

```
bal.tab(f.build("treat", covs), data=lalonde.obs,
        estimand = "ATT", m.threshold =.1,
        disp.var.ratio=FALSE, disp.means=TRUE,
        weights = list(psw = get.w(nsw_ipw),#IPW weights
                       full = get.w(nsw_full))) %>% # Matching weights
  love.plot(abs=FALSE,
            sample.names =
              c("Unadjusted","Inverse Probability Weghting","Optimal full matching")) +
  labs(title= " ") + theme(legend.position = "top")
```

What about the estimated return? As we can see in Figure 4, both models estimate positive and statistically significant retuns to the program of around 2,000 USD. The only noticeably difference are the efficiency gains of optimal full matching with respect to IPW.

```
ipw_out <- lm_robust(f.build("re78", c("treat", covs_raw)), data=lalonde.obs, weights = get.w(nsw_ipw))
full_out <- lm_robust(f.build("re78", c("treat", covs_raw)), data=lalonde.obs, weights = get.w(nsw_full

coeffs_traditional <-
```

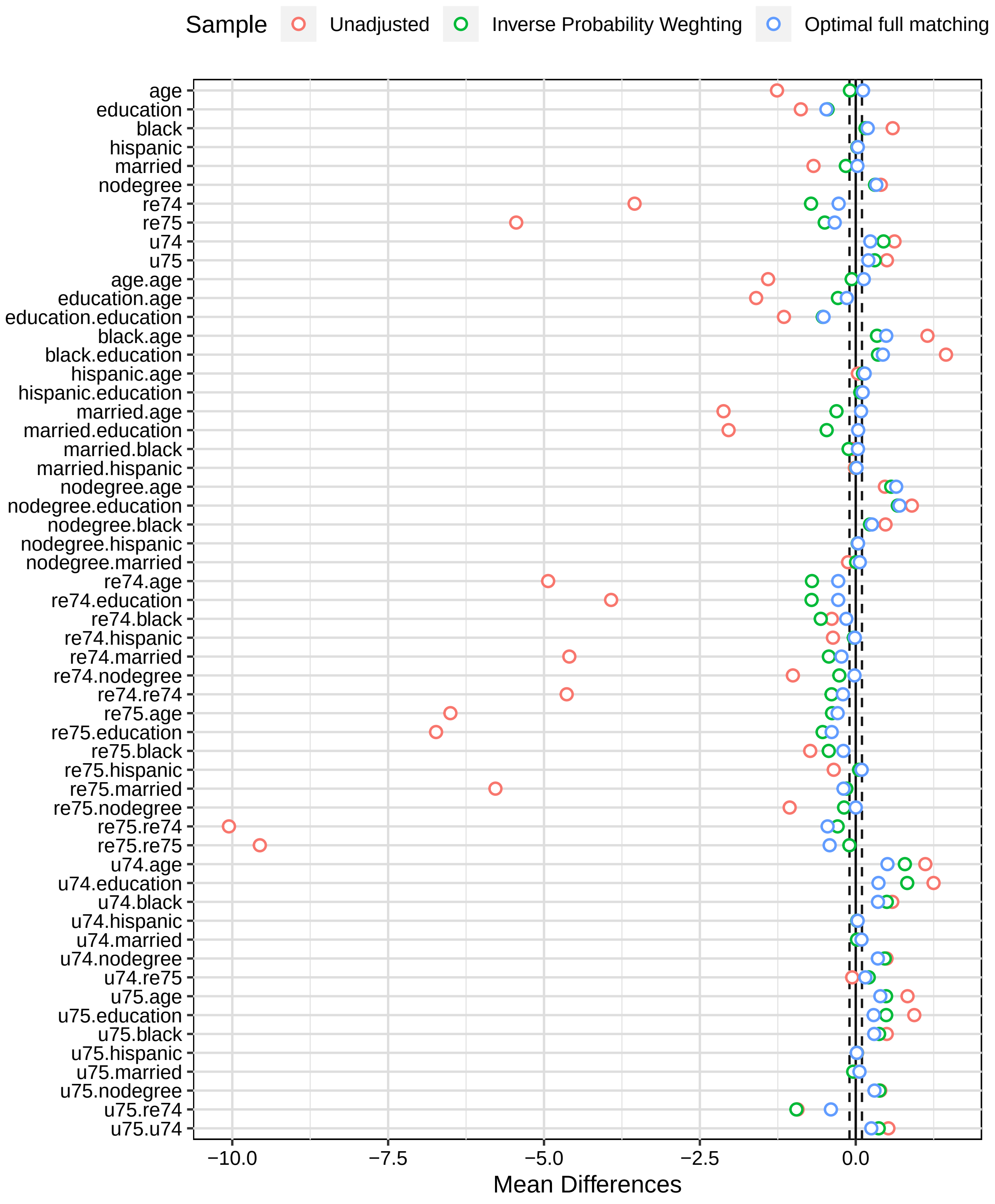

Figure 3: Covariate Balance by Method

```r
  tibble(estimator =
           factor(c("IPW", "OptMatch")),
         coeff = c(ipw_out$coefficients[2],
                   full_out$coefficients[2]),
         se = c(ipw_out$std.error[2],
                full_out$std.error[2]))


# Create the annotated plot
ggplot(coeffs_traditional, aes(x=coeff, y=estimator)) +
  geom_point() +
  geom_errorbarh(aes(x=coeff, y=estimator,
                     xmin=coeff-2*se, xmax=coeff+2*se),
                 height=0) +
  geom_vline(xintercept = 0, linetype="dashed", color="red") +
  annotate("text", x = 2000, y = 1.2, label = "2,017 USD") +
  annotate("text", x = 2000, y = 2.2, label = "2,020 USD") +
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

Example of machine-learning-based method that focuses on modeling the response surface

In this section, I will use traditional random forest (not optimazed for causal inference) to estimate separete regressions for the treated and the controls, and then taking the difference between them, using the `randomForest` package in R [Cutler and Wiener, 2018]. This will not result in a balanced sample or pseudo-population, but it will allow me to obtain estimate returns to the program participation directly comparable with the previos sections.

One important caveate is that there is no pre-canned routine to conduct statistical inference with this procedure, so I am using a very rough approximation using the standard formula for the standard error of a difference in means. It is also important to note that, given our estimand of interest is the ATT, I will use

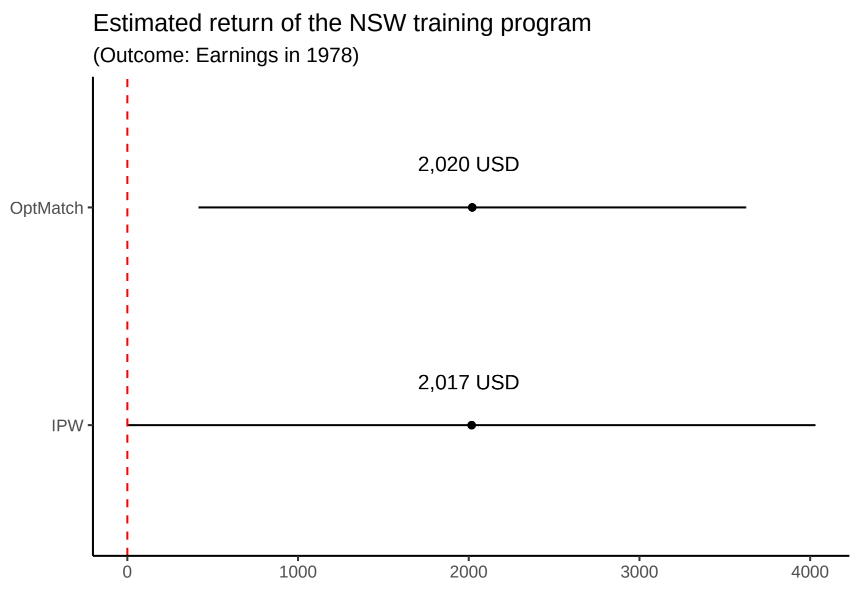

Figure 4: Estimated returns of the training program.

the model fitted to the controls to estimate the expected outcome for the treated units.

The resulting estimate (see Figure 5) of the return is 1,113 USD, somewhat closer to the (difference-in-means) experimental benchmark than the previous models.

```r
library("randomForest")

# Construct the random forest for the controls
rf_control <- randomForest(f.build("re78", covs), # Formula: treatment ~covariates
                           data=lalonde.obs %>% filter(treat==0), # Filter: keep only controls
                           mty=3, ntree=1000) # mty: number of covariate available at each split

# Construct the random forest for the treated
rf_treated <- randomForest(f.build("re78", covs), # Formula: treatment ~covariates
                           data=lalonde.obs %>% filter(treat==1), # Filter: keep only controls
                           mty=3, ntree=1000) # mty: number of covariate available at each split

# Create predictions for treated and controls (using treated data: ATT!)
# This produce is likely invalid inference, but I show it as an approximation

# Predictions for treated units
```

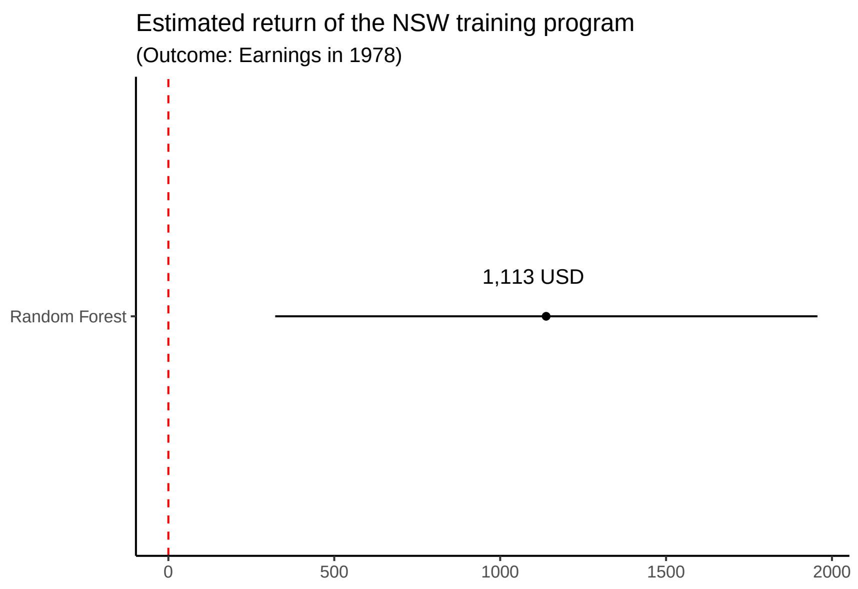

Figure 5: Estimated returns of the training program.

```r
rf_yhat_treat <- predict(rf_treated)
# Prediction for treated units under control (control model)
rf_yhat_control <- predict(rf_control, newdata = lalonde.obs %>% filter(treat==1))
# Model results
rf_out <- tibble(estimator = "Random Forest",
                 coeff = mean(rf_yhat_treat) - mean(rf_yhat_control),
                 se = sqrt(var(rf_yhat_treat)/185 + var(rf_yhat_control)/185))

ggplot(rf_out, aes(x=coeff, y="Random Forest")) +
  geom_point() +
  geom_errorbarh(aes(x=coeff, y="Random Forest",
                     xmin=coeff-2*se, xmax=coeff+2*se),
                 height=0) +
  geom_vline(xintercept = 0, linetype="dashed", color="red") +
  annotate("text", x = 1100, y = 1.1, label = "1,113 USD") +
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

In-sample balance using weighting

I interpret machine-learning-based as meaning one of the two following things: 1) a method that, in contrast to “traditional” methods, seeks to achieve balance (and, if possible, exact mean balance) *in the sample* and not just in expectation, and do so with minimum variance; 2) a [reweighting] method that seeks to achieve balance in a high dimensional projection (such as a kernel transformation) of the covariates, not just on the means of raw covariates.

For this section, I will demonstrate the use of three methods: one IPW model with the propensity score estimated using generalized boosting as showed in [McCaffrey et al., 2004], `entropy balance`, a reweighting method that solves an optimization problem to achieve maximum *in sample* balance; and and a combination of both requirements (propensity score and weighting), the Covariate Balancing Propensity Score [Imai and Ratkovic, 2014], which solves an optimization problem with the double constrain of predicting the treatment status of a unit based on covariates (the *propensity score*), and doing so with the restriction of achieving the maximum *in sample* balance. All these methods are implemented in Greifer [2020]’s `WeightIt` package.

```r
# Estimate the weights
nsw_gbm <-
  weightit(f.build("treat", covs), # Formula: treatment ~ covariates
           data = lalonde.obs, # Declare data
           estimand = "ATT", # Declare estimand
           method = "gbm")
nsw_ebal <-
  weightit(f.build("treat", covs_raw), # Formula: treatment ~ covariates
           data = lalonde.obs, # Declare data
           estimand = "ATT", # Declare estimand
           method = "ebal")
nsw_cbps <-
  weightit(f.build("treat", covs), # Formula: treatment ~ covariates
           data = lalonde.obs, # Declare data
           estimand = "ATT", # Declare estimand
           method = "cbps")
```

As we can see in Figure 6, all the methods achieve very satisfactory balance. There is no considerable difference between the method, except for the notorious balanced sample resulting from `ebal`.

```r
bal.tab(f.build("treat", covs), data=lalonde.obs,
        estimand = "ATT", m.threshold =.1,
        disp.var.ratio=FALSE, disp.means=TRUE,
        weights = list(psw = get.w(nsw_gbm),#IPW weights
                       ebal = get.w(nsw_ebal),
                       cbps = get.w(nsw_cbps))) %>% # Matching weights
  love.plot(abs=FALSE,
            sample.names =
              c("Unadjusted","Generalized Boosting","Entropy Balance", "Covariate Balancing PS")) +
  labs(title= " ") + theme(legend.position = "top")
```

In terms of the outcome estimation, in Figure 7 we can see that both methods relying on Propensity Score estimation calculate a return of around 2,100 USD, while the reweighted sample using entropy balance overestimate the returns of the program, calculating an effect of around 2,400 USD. This is somewhat surprising and it should rise awareness of the relation between balance and unbiasedness, because at least in this case the more balanced sample is at the same time the one that produces the further point estimate in comparison to the experimental benchmark. On the other hand, the CBPS method produced results relatively closer to the experimental benchmark, but substantially less precise.

```r
gbm_out <- lm_robust(f.build("re78", c("treat", covs_raw)), data=lalonde.obs, weights = get.w(nsw_gbm))
ebal_out <- lm_robust(f.build("re78", c("treat", covs_raw)), data=lalonde.obs, weights = get.w(nsw_ebal
cbps_out <- lm_robust(f.build("re78", c("treat", covs_raw)), data=lalonde.obs, weights = get.w(nsw_cbps

coeffs_ml <-
  tibble(estimator =
           factor(c("Generalized Boosting","Entropy Balance", "Covariate Balancing PS")),
         coeff = c(gbm_out$coefficients[2],
                   ebal_out$coefficients[2],
                   cbps_out$coefficients[2]),
         se = c(gbm_out$std.error[2],
                ebal_out$std.error[2],
                cbps_out$std.error[2]))
```

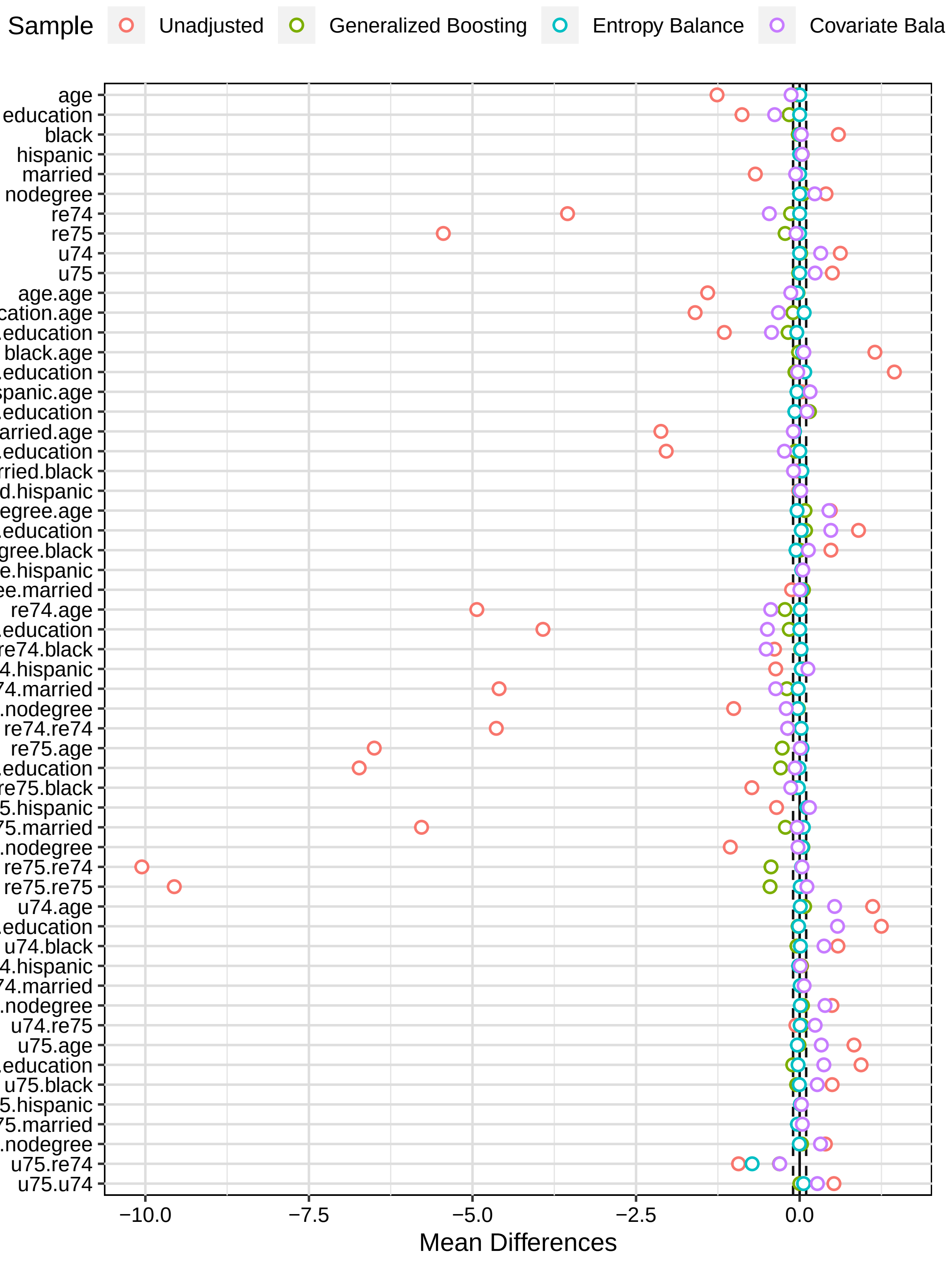

Figure 6: Covariate Balance by Method

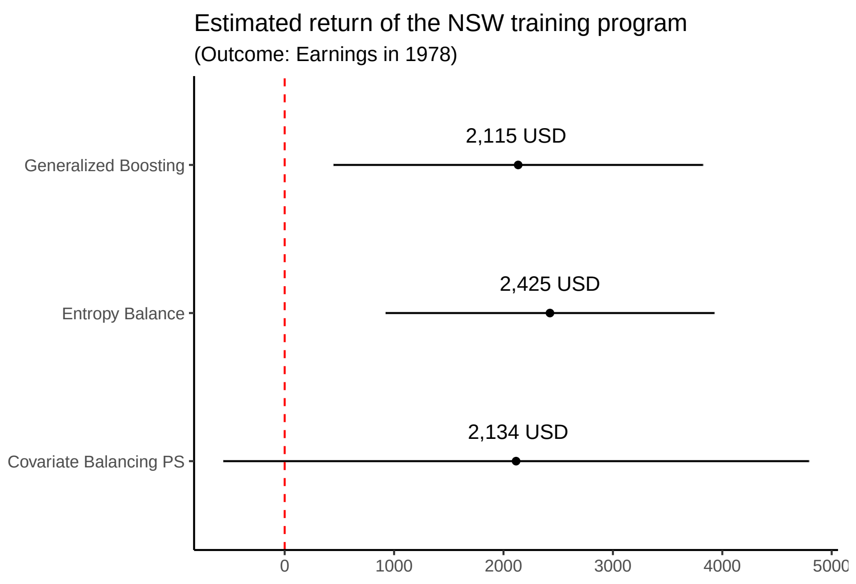

Figure 7: Estimated returns of the training program.

```r
# Create the annotated plot
ggplot(coeffs_ml, aes(x=coeff, y=estimator)) +
  geom_point() +
  geom_errorbarh(aes(x=coeff, y=estimator,
                     xmin=coeff-2*se, xmax=coeff+2*se),
                 height=0) +
  geom_vline(xintercept = 0, linetype="dashed", color="red") +
  annotate("text", x = 2134, y = 1.2, label = "2,134 USD") +
  annotate("text", x = 2425, y = 2.2, label = "2,425 USD") +
  annotate("text", x = 2115, y = 3.2, label = "2,115 USD") +
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

Comparing the results of these different methods using the appropriate metrics.

Below, I use two metrics to compare the performance of different methods: 1) the covariate balance they are able to achieve; 2) how close their estimate of the ATT comes to the experimental benchmark. This is just a combination of the previous graphs to show in one place all the results and facilitate the comparison.

First, we can see in Figure 8 that contemporary methods tend to outperform the traditional IPW in terms of achieving balance. The difference is not so clear with respect to the optimal full matching implementation, so we can see that it is still a state-of-the-art approach[3]. Among the modern version, the only one that systematically outperfoms the rest in terms of balance it Entropy Balancing.

```
bal.tab(f.build("treat", covs), data=lalonde.obs,
        estimand = "ATT", m.threshold =.1,
        disp.var.ratio=FALSE, disp.means=TRUE,
        weights = list(psw = get.w(nsw_ipw), # IPW weights
                       full = get.w(nsw_full), # Full matchin weights
                       gbm = get.w(nsw_gbm), # Generalized boosting model weights
                       ebal = get.w(nsw_ebal), # Entropy Balance weights
                       cbps = get.w(nsw_cbps))) %>% # CBPS weights
  love.plot(abs=TRUE,
            sample.names =
              c("Unadjusted","Inverse Probability Weghting","Optimal full matching",
                "Generalized Boosting","Entropy Balance","Covariate Balancing PS")) +
  labs(title= " ") + theme(legend.position = "top")
```

In terms of effect estimation, we can see a consolidated version in in Figure 9. Experimental benchmarks are in the vertical lines. On the good side, all the estimators produce confidence intervals that cover the true parameters (both the difference-in-means and the Lin estimator in the experimental sample). Putting aside the CI for the random forest[4], either matching or direct balance tend to produce more precise estimates, while weighting based on the propensity score tend to be accompanied by more variance.

On the "negative" side, and if we look at the point estimates, we can see that most methods but one (random forest) overestimate the benefit of the program, while random forest (the outcome regression version) underestimate it. Therefore, it is possible to think that a combination of both approaches can improve into these methods. This is the focus of the last question.

```
rf_out$estimator <- as.factor(rf_out$estimator)
coeffs_all <- rbind(coeffs_traditional, coeffs_ml, rf_out)
```

[3]This will not be true in general for other matching approches, which tend to be outperformed by optimal full matching
[4]As I declare above, those confidence intervals should be taken with a bag of salt.

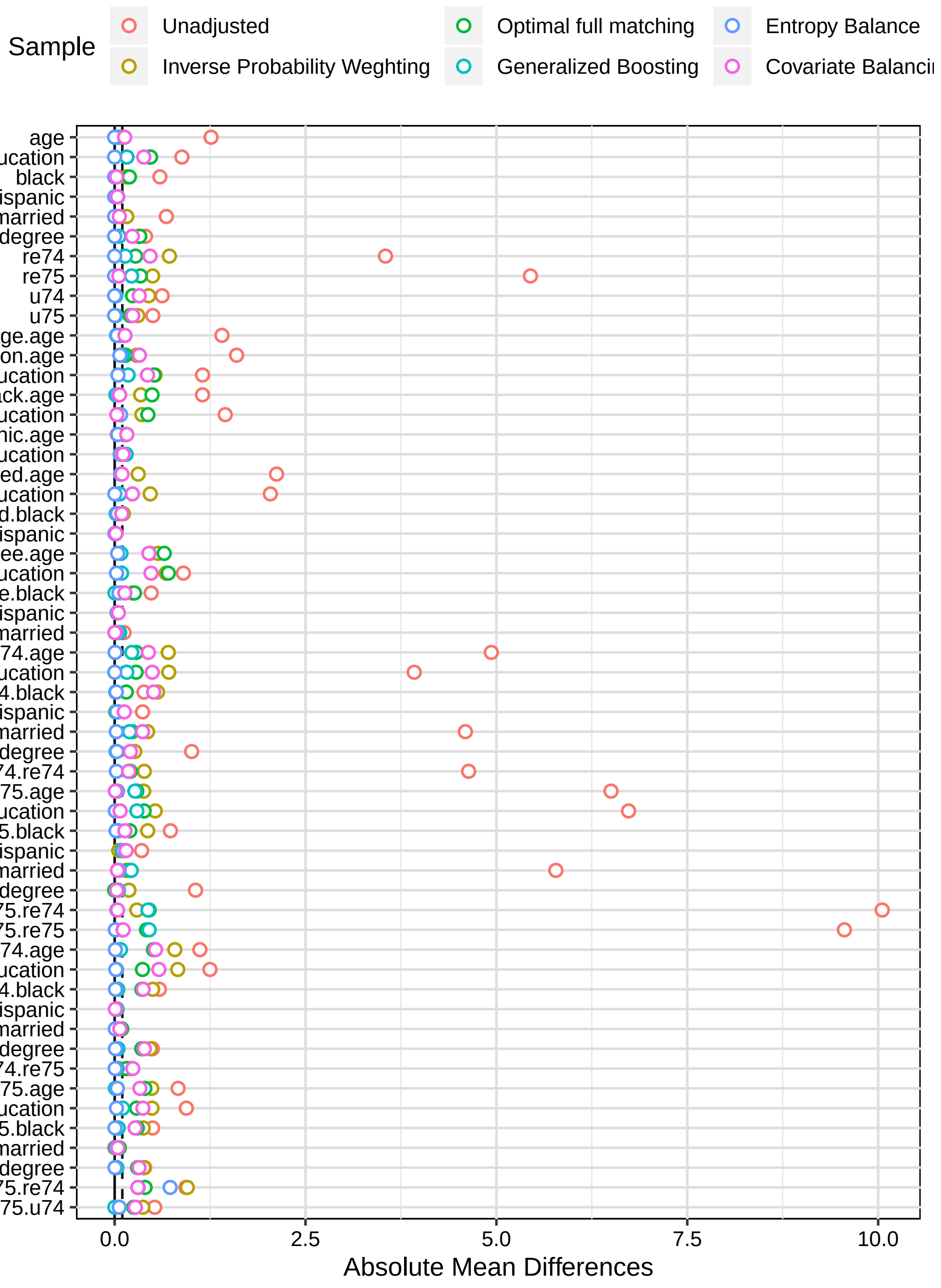

Figure 8: Covariate Balance by Method

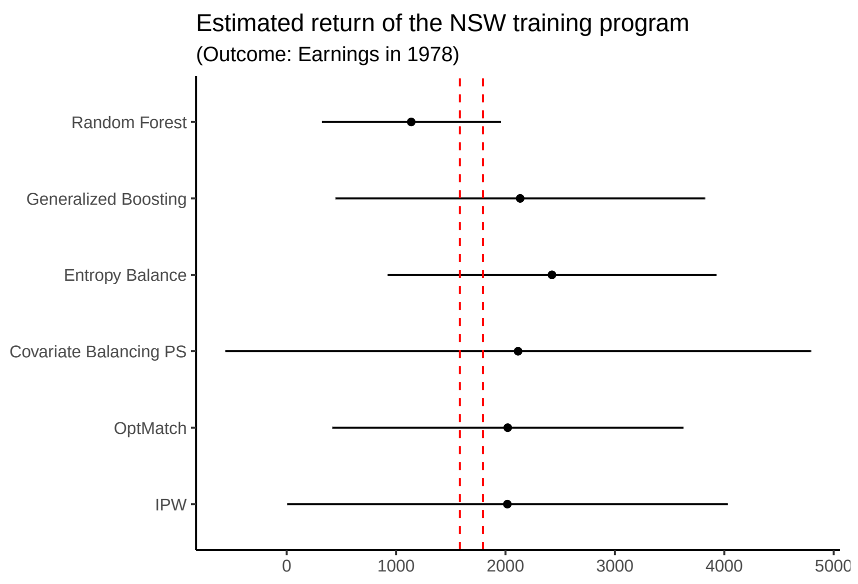

Figure 9: Estimated returns of the training program.

```r
# Create the annotated plot
ggplot(coeffs_all, aes(x=coeff, y=estimator)) +
  geom_point() +
  geom_errorbarh(aes(x=coeff, y=estimator,
                     xmin=coeff-2*se, xmax=coeff+2*se),
                 height=0) +
  geom_vline(xintercept = 1794, linetype="dashed", color="red") +
  geom_vline(xintercept = 1583, linetype="dashed", color="red") +
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

A crappy doubly robust estimator

Doubly-robust methods, also known as the augmented propensity score weighting estimator [Glynn and Quinn, 2010/ed], are a combination of regression modeling and weighting. It is called doubly robust because it is sufficient for one of the piece (propensity score, outcome model) to be correclty specified to have an unbiased estimator of the causal effect. More agnostically, one can say that the errors in the models multiply each other, so even if no piece is perfectly estimate, we can in general observe a better behavior of these type of estimators.

In this section, I will implement a doubly robust estimator "by hand", combining the weights produced by Entropy Balance, and the predictions of $\mathbb{E}[Y_{0i}|D = 0, X]$ obtained from the Random Forest model for the controls.

Given that the balance in this estimator is the same already shown for Entropy Balance, I do not show it again (but remember that it is the best balance achieved among weighted samples). Figure 10 demonstrate that, among all estimators, the doubly robust version just described here generates the closest point estimate with respect to the experimental benchmark, and its confidence interval includes the true parameter. This is a well-known result in the literature, and it is in general adivsable to exploit both weighting and regression when attempting to estimate treatment effects from observational data.

```r
# Using RF model for the outcome regression
y1_hat <- predict(rf_treated, newdata = lalonde.obs) # Used for ATE
y0_hat <- predict(rf_control, newdata = lalonde.obs) # Used fot ATT

# Using ebal for weighting, adjusting for predicted Y(0) function
dr_out <- lm_robust(re78 ~ treat + y0_hat, data=lalonde.obs, weights = get.w(nsw_ebal))

coeffs_dr <- tibble(estimator = "Doubly Robust Estimator",
         coeff = dr_out$coefficients[2],
         se = dr_out$std.error[2])

coeffs_all <- rbind(coeffs_all, coeffs_dr)

# Create the annotated plot
ggplot(coeffs_all, aes(x=coeff, y=estimator)) +
  geom_point() +
  geom_errorbarh(aes(x=coeff, y=estimator,
                     xmin=coeff-2*se, xmax=coeff+2*se),
                 height=0) +
  geom_vline(xintercept = 1794, linetype="dashed", color="red") +
```

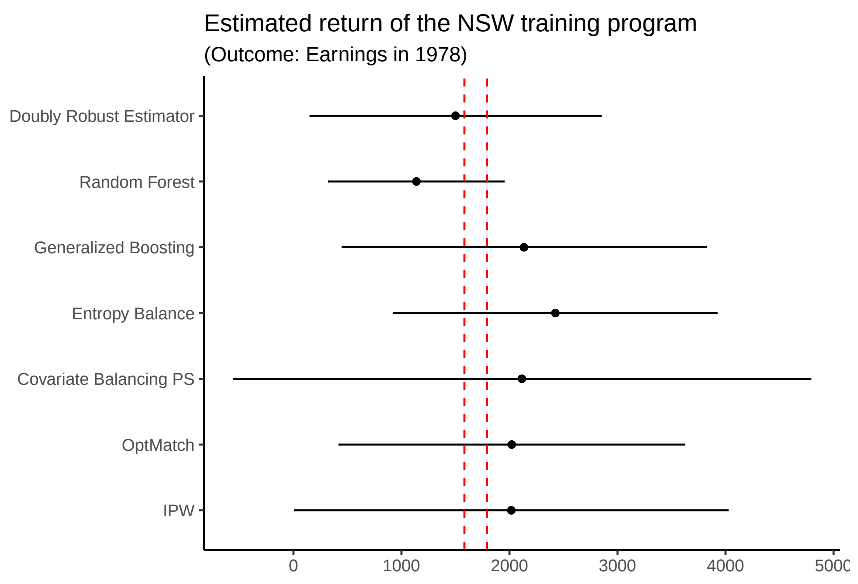

Figure 10: Estimated returns of the training program.

```
  geom_vline(xintercept = 1583, linetype="dashed", color="red") +
  theme_classic() +
  labs(title = "Estimated return of the NSW training program",
       subtitle = "(Outcome: Earnings in 1978)",
       x = " ", y= " ")
```

Exercise: Difference-in-difference by hand

Now, try something yourself. Another set of assumptions commonly invoked in causal inference in observational settings, when you have longitudinal outcome data, is known as parallel trends (aka difference-in-differences). Formally, the assumption can be written as:

$$E[Y_0(t=1) - Y_0(t=0)|D=0] = E[Y_0(t=1) - Y_0(t=0)|D=1]$$

In plan terms, this means that we believe that, if no treatment happens for any group ($D = 1$ and $D = 0$), their trends over time ($Y_0(t = 1) - Y_0(t = 0)$) would be the same. As usual, this is untestable, because what we have is that no group is treated in $t = 0$, but one group receives treatment in $t = 1$ while the other remains untreated.

The diff-in-diff estimator is constructed by a double difference (hence the name):

$$\Big[\underbrace{E(Y|D=1,t=1)-E(Y|D=0,t=1)}_{\text{Difference 1}}\Big]-\Big[\underbrace{E(Y|D=1,t=0)-E(Y|D=0,t=0)}_{\text{Difference 2}}\Big]$$

The first difference is the treated minus controls average in the post-treatment period, and the second difference is the treated minus controls average in the pre-treatment period. In a sense, we are trying to discount "fixed differences" between groups.

Another (equivalent) way to look at this is:

$$\Big[\underbrace{E(Y|D=1,t=1)-E(Y|D=1,t=0)}_{\text{Difference 1}}\Big]-\Big[\underbrace{E(Y|D=0,t=1)-E(Y|D=0,t=0)}_{\text{Difference 2}}\Big]$$

Here, the first difference is the post period minus pre period difference for the group that becomes treated, while the second difference is the post minus pre difference for the group that remains untreated. This makes more explicit that diff-in-diff is a pre-post comparison "adjusted" for possible time trends that the second difference is capturing.

In the Lalonde dataset, we have the outcome `re78` (earnings in 1978) and previous measures of the outcome `re75` (earnings in 1975). Now, calculate the difference in difference "by hand" (using R, of course) both ways, and compare your results with the experimental benchmark. Enjoy!

References

Fortran original by Leo Breiman and Adele Cutler and R. port by Andy Liaw and Matthew Wiener. randomForest: Breiman and Cutler's Random Forests for Classification and Regression, March 2018.

Adam N. Glynn and Kevin M. Quinn. An Introduction to the Augmented Inverse Propensity Weighted Estimator. *Political Analysis*, 18(1):36–56, 2010/ed. ISSN 1047-1987, 1476-4989. doi: 10.1093/pan/mpp036.

Noah Greifer. WeightIt: Weighting for Covariate Balance in Observational Studies, February 2020.

Jens Hainmueller. Entropy Balancing for Causal Effects: A Multivariate Reweighting Method to Produce Balanced Samples in Observational Studies. *Political Analysis*, 20(1):25–46, 2012. ISSN 1047-1987, 1476-4989. doi: 10.1093/pan/mpr025.

Ben B. Hansen, Mark Fredrickson, Josh Buckner, Josh Errickson, Adam Rauh and Peter Solenberger, and with embedded Fortran code due to Dimitri P. Bertsekas and Paul Tseng. Optmatch: Functions for Optimal Matching, December 2019.

Kosuke Imai and Marc Ratkovic. Covariate balancing propensity score. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):243–263, 2014. ISSN 1467-9868. doi: 10.1111/rssb.12027.

Robert J Lalonde. Evaluating the Econometric Evaluations of Training Programs with Experimental Data. *The American Economic Review*, 76(4):604–620, 1986.

Daniel F. McCaffrey, Greg Ridgeway, and Andrew R. Morral. Propensity score estimation with boosted regression for evaluating causal effects in observational studies. *Psychological Methods*, 9(4):403–425, December 2004. ISSN 1082-989X. doi: 10.1037/1082-989X.9.4.403.