Matching and Human Capital

Alex Alekseev

```r
library(tidyverse)
library(Matching)
```

Intro

Let's replicate the analysis from the "Example: Human Capital Revisited". First let's generate the data according to the tables in the example.

The distribution of ability (S) is

S	$\mathbb{P}(S)$
1	0.44
2	0.24
3	0.32

The conditional distribution of college D given ability is

	$S = 1$	$S = 2$	$S = 3$
$\mathbb{P}(D = 0 \mid S)$	9/11	1/2	3/8
$\mathbb{P}(D = 1 \mid S)$	2/11	1/2	5/8

The mean income conditional on ability and college is

	$\mathbb{E}[Y^0 \mid D = 0, S]$	$\mathbb{E}[Y^1 \mid D = 1, S]$	$\mathbb{E}[\delta \mid S]$
$S = 1$	2	4	2
$S = 2$	6	8	2
$S = 3$	10	14	4

The code below generates the data according to these rules.

```r
nobs <- 10000
set.seed(42)
df <-
  tibble(
    ability =
      sample(
        c(1, 2, 3), prob = c(0.44, 0.24, 0.32)
        , size = nobs, replace = T
      )
    , educ =
      1*(
        (runif(nobs) <= 2/11)*(ability == 1)
        + (runif(nobs) <= 1/2)*(ability == 2)
        + (runif(nobs) <= 5/8)*(ability == 3)
      )
    , income =
      2 + 2*educ + (ability - 1)*4
    + 2*educ*(ability == 3) + rnorm(nobs)
  )
```

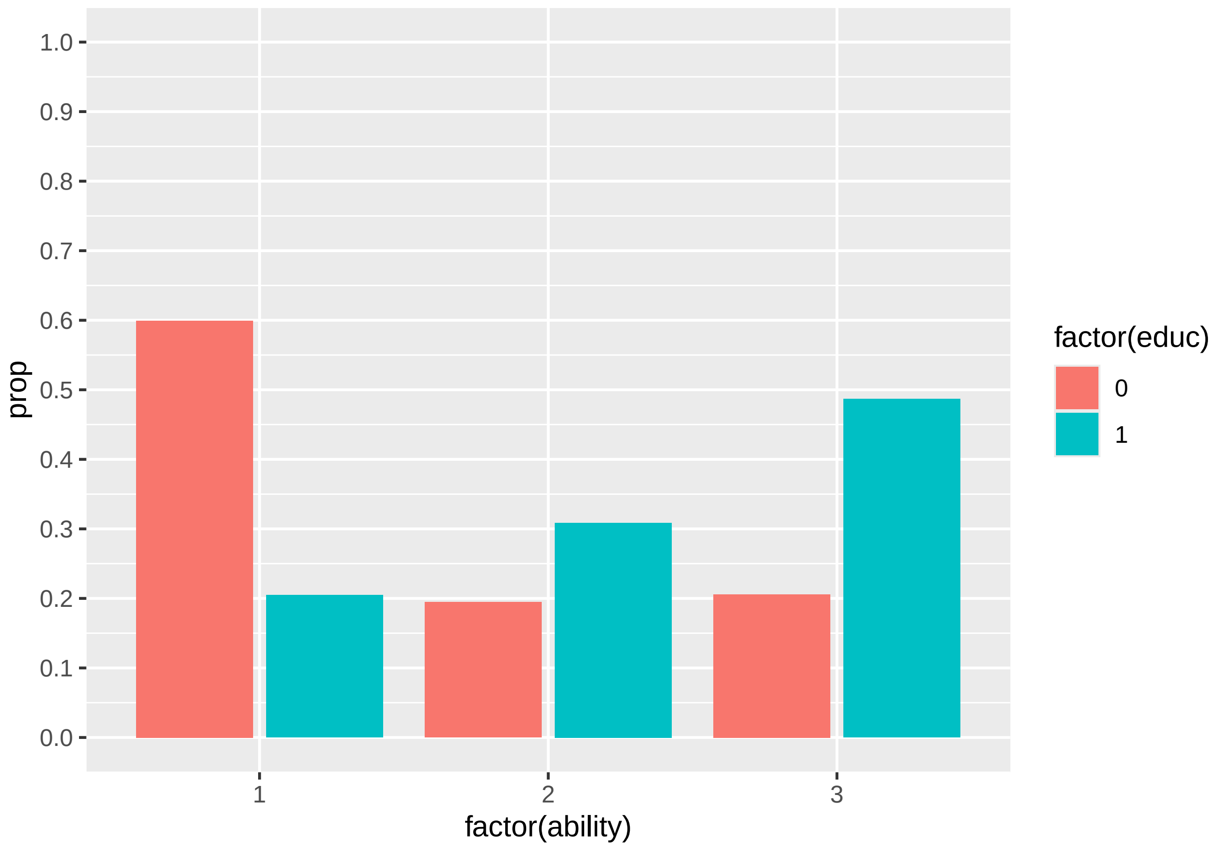
Let's verify that the mean income conditional on ability and college is the same as in our table.

```
df %>%
  group_by(educ, ability) %>%
  summarise(mean_income = mean(income)) %>%
  pivot_wider(
    names_from = educ, values_from = mean_income, names_prefix = "educ_"
  )
```

ability	educ_0	educ_1
1	2.010063	4.035757
2	5.986093	7.965154
3	9.991611	14.043311

Now let's have a look at the distribution of ability in the treatment and control groups.

```
ggplot(df, aes(factor(ability), group = factor(educ))) +
  stat_count(
    aes(y = after_stat(prop), fill = factor(educ)), position = "dodge2"
  ) +
  scale_y_continuous(limits = c(0, 1), n.breaks = 10)
```



The graph shows that the people who went to college have a different distribution of ability than the people who did not go to college. We will have to fix that. Notice that the graph matches the numbers for the conditional distribution of S given D that we derived by hand in the example.

Matching

The benefit of matching is that we can easily compute the ATT, ATU, and ATE, while regression only gives us a single treatment effect. The downside of matching is that it requires a lot of data and can be much slower.

We will now compute each of the three treatment effects using the `Match` function from the `Matching` package.

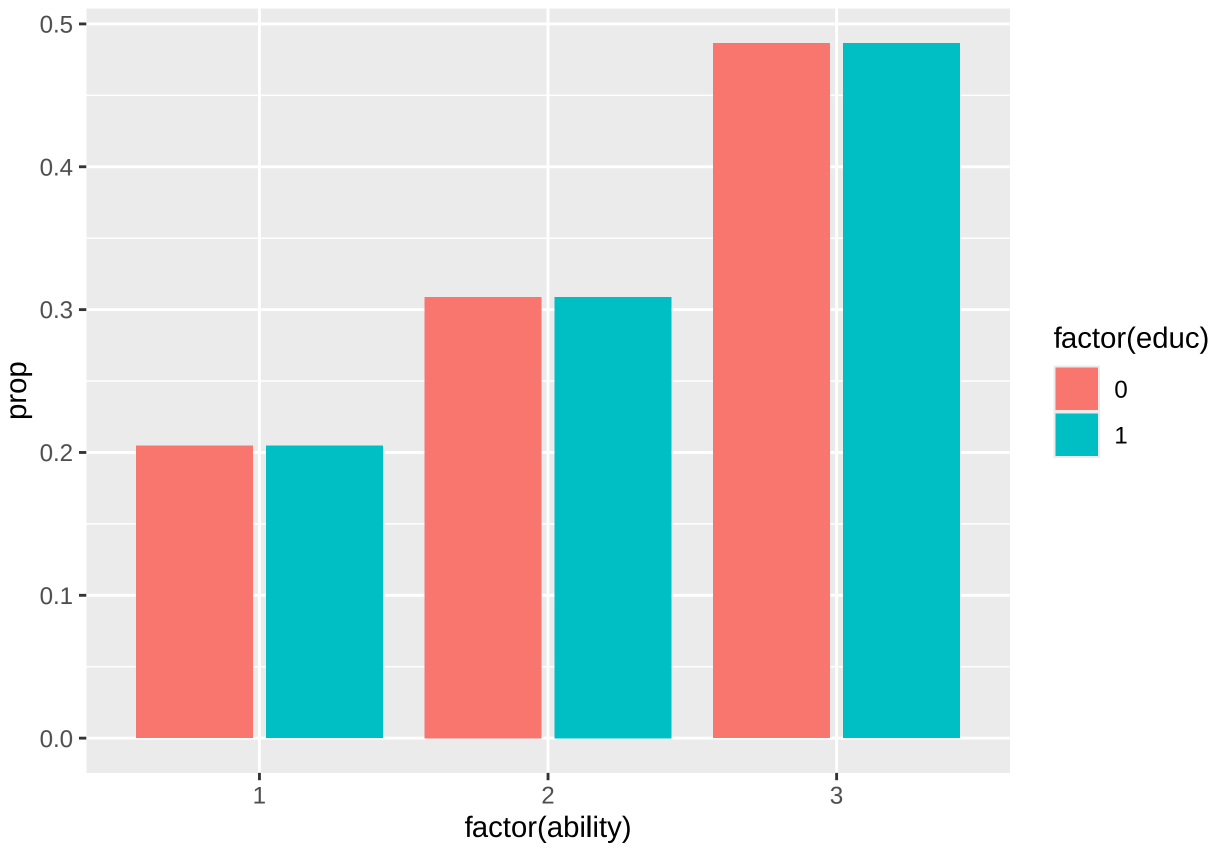
```r
M_att <- Match(
  Y = df$income, Tr = df$educ, X = df$ability
  , estimand = "ATT"
  , exact = T
  , ties = F
)

summary(M_att)
```

```
##
## Estimate...  2.978
## SE.........  0.027869
## T-stat.....  106.86
## p.val......  < 2.22e-16
##
## Original number of observations..............  10000
## Original number of treated obs...............  4051
## Matched number of observations...............  4051
## Matched number of observations  (unweighted).  4051
##
## Number of obs dropped by 'exact' or 'caliper'  0
```

This is very close to the true ATT of 3. Let's see what matching did to the distribution of ability in the matched treatment and control groups.

```
bind_rows(df[M_att$index.treated, ], df[M_att$index.control, ]) %>%
  ggplot(aes(factor(ability), group = factor(educ))) +
  stat_count(
    aes(y = after_stat(prop), fill = factor(educ)), position = "dodge2"
  )
```



Perfect balance! Specifically, notice that the matched control group has the exact same distribution of ability as the treatment group.

Let's repeat the analysis for the ATU. Notice that the `Match` function refers to the ATU as *ATC*.

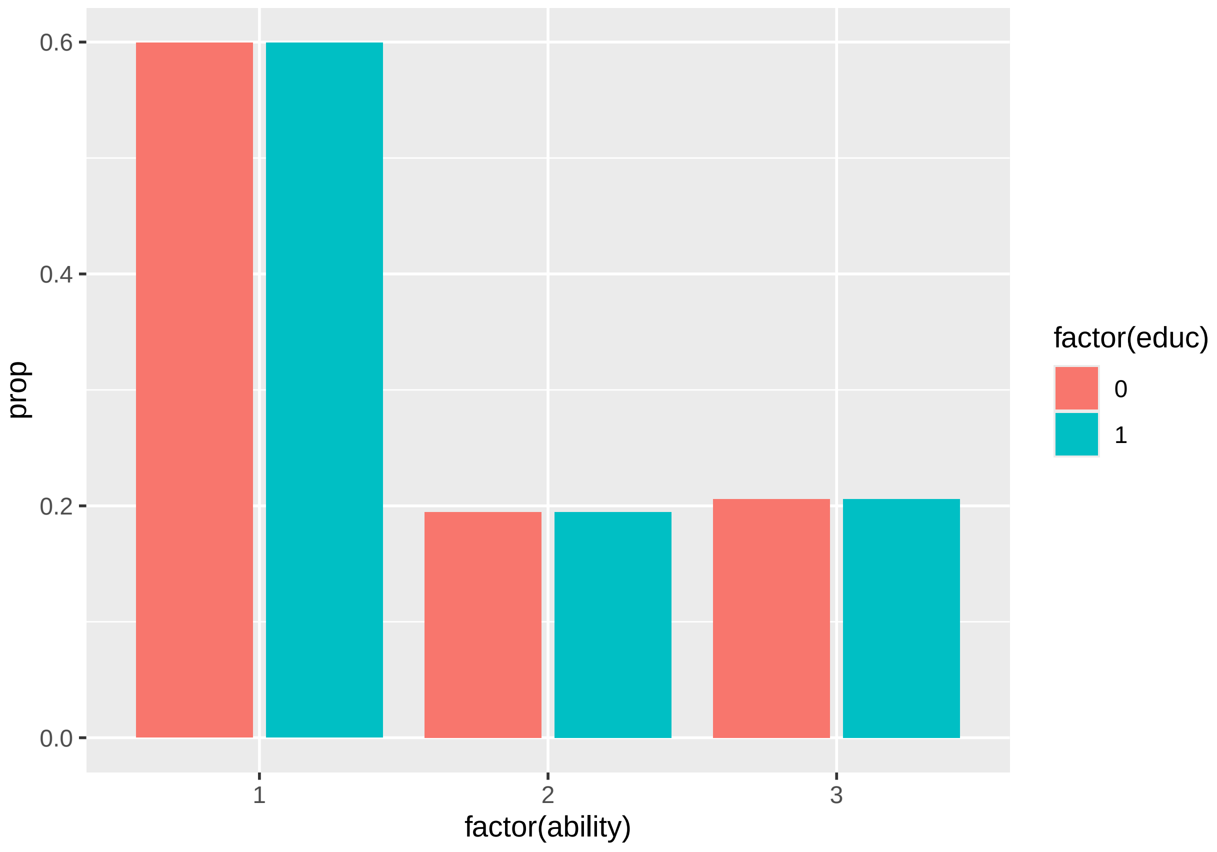
```
M_atu <- Match(
  Y = df$income, Tr = df$educ, X = df$ability
  , estimand = "ATC"
  , exact = T
  , ties = F
)

summary(M_atu)
```

```
##
## Estimate...  2.4208
## SE.........  0.021382
## T-stat.....  113.21
## p.val......  < 2.22e-16
##
## Original number of observations..............  10000
## Original number of control obs...............  5949
## Matched number of observations...............  5949
## Matched number of observations  (unweighted).  5949
##
## Number of obs dropped by 'exact' or 'caliper'  0
```

Again, pretty close to the true value of ATU = 2.4. Let's check the covariate balance.

```
bind_rows(df[M_atu$index.treated, ], df[M_atu$index.control, ]) %>%
  ggplot(aes(factor(ability), group = factor(educ)) +
  stat_count(
    aes(y = after_stat(prop), fill = factor(educ)), position = "dodge2"
  )
```



Again, perfect balance. Specifically, the distribution of ability in the matched treated group is exactly the same as in the control group.

Finally, let's do the ATE.

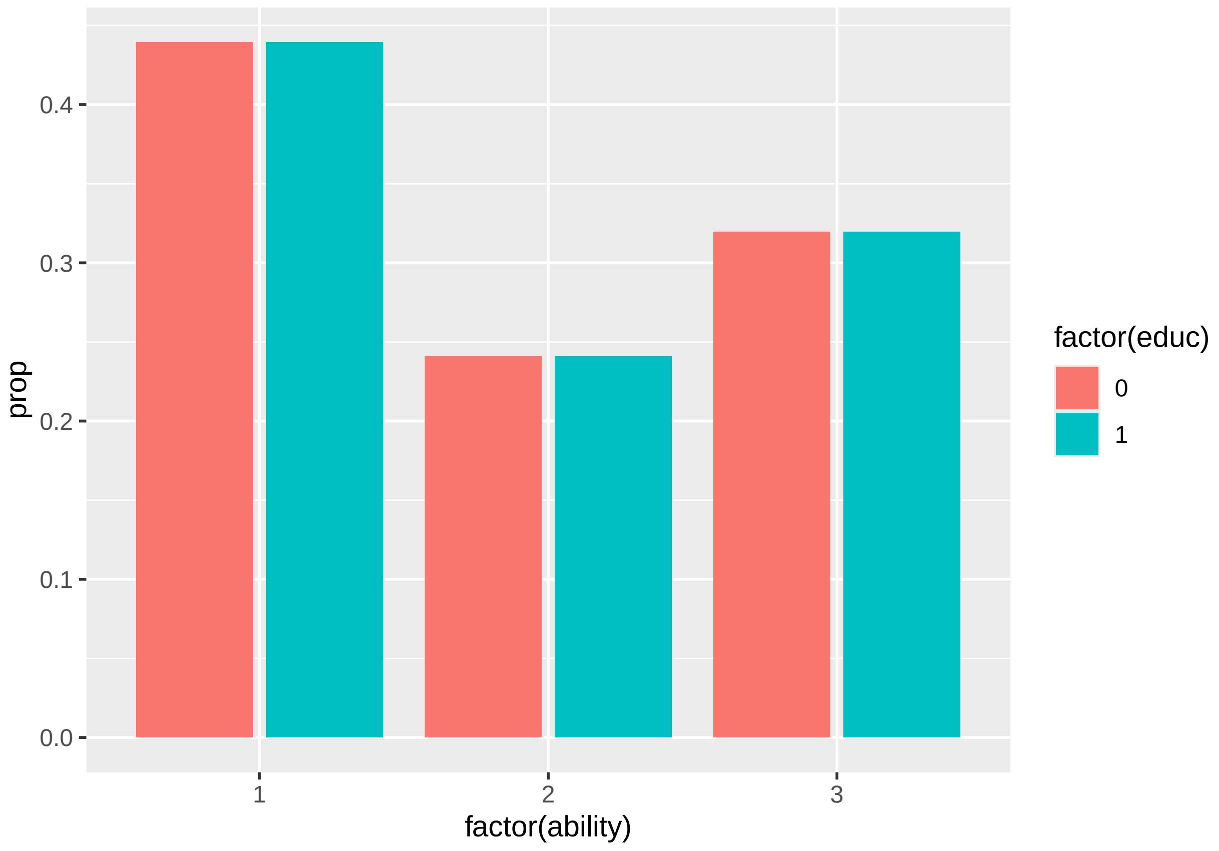
```r
M_ate <- Match(
  Y = df$income, Tr = df$educ, X = df$ability
  , estimand = "ATE"
  , exact = T
  , ties = F
)

summary(M_ate)
```

```
##
## Estimate...  2.671
## SE.........  0.017296
## T-stat.....  154.43
## p.val......  < 2.22e-16
##
## Original number of observations..............  10000
## Original number of treated obs...............  4051
## Matched number of observations...............  10000
## Matched number of observations  (unweighted).  10000
##
## Number of obs dropped by 'exact' or 'caliper'  0
```

Pretty close to the true value of ATE $= 2.64$. Let's check the covariate balance.

```r
bind_rows(df[M_ate$index.treated, ], df[M_ate$index.control, ]) %>%
  ggplot(aes(factor(ability), group = factor(educ))) +
  stat_count(
    aes(y = after_stat(prop), fill = factor(educ)), position = "dodge2"
  )
```



Perfect balance. Now both groups have the distribution of ability that is exactly the same as the unconditional distribution of ability in the full sample.

A more sophisticated way of checking for covariate balance would be to use the built-in function for this. For example,

```
MatchBalance(educ ~ ability, data = df, match.out = M_att)
```

```
##
## ***** (V1) ability *****
##                        Before Matching        After Matching
## mean treatment........     2.2819             2.2819
## mean control..........     1.6065             2.2819
## std mean diff.........     86.346                  0
##
## mean raw eQQ diff.....    0.67564                  0
## med  raw eQQ diff.....          1                  0
## max  raw eQQ diff.....          2                  0
##
## mean eCDF diff........    0.22514                  0
## med  eCDF diff........    0.28063                  0
## max  eCDF diff........    0.39479                  0
##
## var ratio (Tr/Co).....    0.94046                  1
## T-test p-value........ < 2.22e-16                  1
## KS Bootstrap p-value.. < 2.22e-16                  1
## KS Naive p-value...... < 2.22e-16                  1
## KS Statistic..........    0.39479            0
```

Propensity score

Instead of matching on covariates directly, we can use the propensity score. For this to work, we first have to estimate it. We can do this by running a simple linear regression of education on ability as a factor. We then save the predicted propensity scores.

```
ps_reg <- lm(educ ~ factor(ability), data = df)
ps <- predict(ps_reg)
head(ps)
```

```
##         1         2         3         4         5         6
## 0.5193026 0.5193026 0.1886234 0.5193026 0.6167084 0.6167084
```

To match on the propensity score, we need to change the `X` argument in the `Match` function.

```
M <- Match(
  Y = df$income, Tr = df$educ, X = ps
  , estimand = "ATT"
  , exact = T
  , ties = F
)

summary(M)
```

```
##
## Estimate...  2.9935
## SE.........  0.027782
## T-stat.....  107.75
## p.val......  < 2.22e-16
##
## Original number of observations..............  10000
## Original number of treated obs...............  4051
## Matched number of observations...............  4051
## Matched number of observations  (unweighted).  4051
##
## Number of obs dropped by 'exact' or 'caliper'  0
```

The estimate is pretty close to 3, although the number is slightly different than the one we got by matching on ability itself. We can obtain other treatment effects in a similar fashion.

Weighting

Another way to use the propensity score is to construct weights. Recall the expression for the ATT estimate.

$$\widehat{ATT} = \frac{1}{n^1}\left(\sum_{i\in I} y_i - \sum_{j\in J}\frac{p_j}{1-p_j}y_j\right)$$

Now we just need to implement it in our data. First, we will save the indices of observations corresponding to the treatment and control observations. They correspond to I and J in the formula.

```r
treatment_i <- seq_len(nobs)[df$educ == 1]
control_j <- seq_len(nobs)[df$educ == 0]
```

Now we will create a weighting variable `weight_att` and add it to our dataset. The weights are equal 1 for the treatment observations and $p_j/(1-p_j)$ for the control observations. We can then plug everything into the formula to get the estimate of the ATT.

```r
df$weight_att[treatment_i] <- 1
df$weight_att[control_j] <- ps[control_j]/(1 - ps[control_j])

1/length(treatment_i) * (
  sum(df$income[treatment_i]) -
    sum(df$income[control_j]*df$weight_att[control_j])
)
```

```
## [1] 2.997039
```

That looks pretty close to the true effect.

To estimate the ATU, we need to repeat the previous steps but use a different weighting scheme.

```r
df$weight_atu[treatment_i] <- (1 - ps[treatment_i])/ps[treatment_i]
df$weight_atu[control_j] <- 1

1/length(control_j) * (
  sum(df$income[treatment_i]*df$weight_atu[treatment_i]) -
    sum(df$income[control_j])
)
```

```
## [1] 2.433806
```

Again, pretty close to the true effect.

And finally, we can also compute the ATE.

```r
df$weight_ate[treatment_i] <- 1/ps[treatment_i]
df$weight_ate[control_j] <- 1/(1 - ps[control_j])

1/nobs * (
  sum(df$income[treatment_i]*df$weight_ate[treatment_i]) -
    sum(df$income[control_j]*df$weight_ate[control_j])
)
```

```
## [1] 2.661972
```

Regression

Let's look at what we would get from a regression. We will run a naive regression of outcome on treatment, and a more sophisticated regression where we include ability as control variable. We will use a fully flexible coding for ability and treat it as a factor variable.

```r
reg1 <- lm(income ~ educ, data = df)
reg2 <- lm(income ~ educ + factor(ability), data = df)

coef(reg1)[2]
```

```
##     educ
## 5.690794
```

```r
coef(reg2)[2]
```

```
##     educ
## 2.766061
```

The coefficient from the first regression is unsurprisingly biased. But notice what we get from the second, supposedly better, regression. The result is pretty close to the ATE, but not quite there.