Pset6

2024-02-14

In this code block, we import the data, as well as installing any necessary packages.

```
source("./Functions.R")

install_packages_if_needed(c("utils"))
library(tidyr)

#Import the csv files
dt_psid <- data.table::as.data.table(utils::read.delim(file = "nswpsid.csv",
                                    sep = ","))
```

First, we need the Logit-based P-Score, based on the formula in PSet 5:

```
#mutate the data table to add additional variables:
dt_psid <-dt_psid %>%
  dplyr::mutate(.data=dt_psid,
                agesq = age**2,
                edusq = edu**2,
                re74sq = re74**2,
                re75sq = re75**2,
                u74black = u74*black)


#Write the formula (as in Pset 5):
pscore_formula <- as.formula("treat ~ age + agesq +edu + edusq + married + nodegree + black +
hisp + re74 + re75 + re74sq + re75sq + u74black")

#mle estimation
mle <- stats::glm(pscore_formula, family = binomial( ), data = dt_psid)

#prediction
p_logit <- stats::predict(mle, type = "response")

#Store in dt
dt_psid <-dt_psid %>%
  dplyr::mutate(.data=dt_psid,
                p_logit = p_logit)

summary(p_logit)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0000341 0.0006388 0.0691589 0.0109155 0.9748754
```

Q1

```
#This gives the summary of the p-scores of the control and treatment groups:
dt_psid %>%
    dplyr::group_by(treat) %>%
    dplyr::summarise_at(.var=c('p_logit'), .funs = c(max = max, min=min))
```

```
## # A tibble: 2 × 3
##   treat   max      min
##   <int> <dbl>    <dbl>
## 1     0 0.974 4.49e-11
## 2     1 0.975 6.53e- 4
```

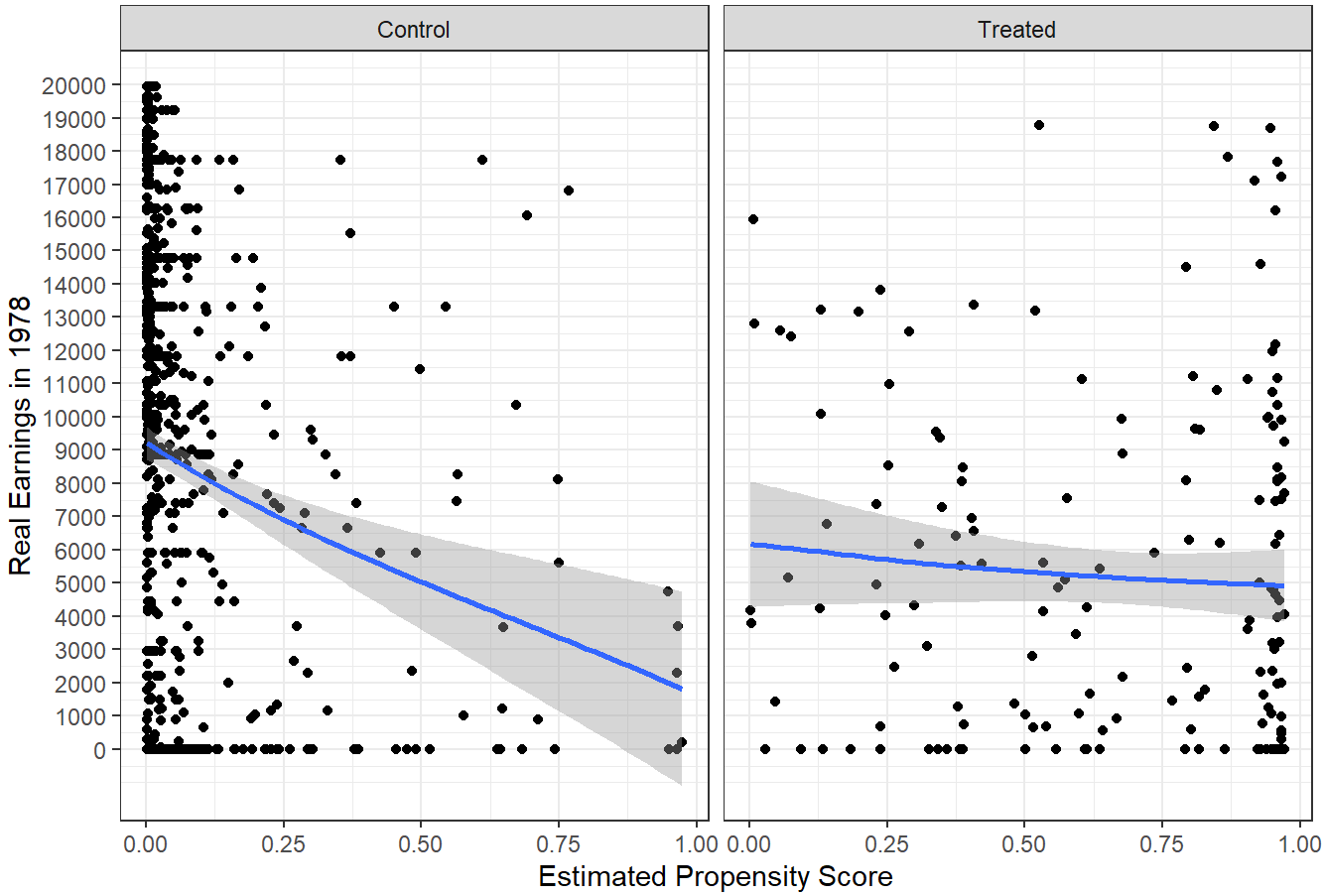
```
#Now find the common support in treat and control groups:
upper_p <- min(by(data = dt_psid$p_logit, INDICES = dt_psid$treat, FUN = max))
lower_p <- max(by(data = dt_psid$p_logit, INDICES = dt_psid$treat, FUN = min))

#Now we trim the data set using these bounds:
psid_trimmed <- dplyr::filter(.data = dt_psid, p_logit >= lower_p & p_logit <= upper_p)
```

Q2 a. This is from the problem set:

```
# Draw scatter plot of post-intervention earnings and the Logit-based pscore, by group;
# overlay smooth local regression line.
plot_df <- psid_trimmed %>%
  dplyr::filter(re78 < 20000) %>% # drop outliers for plotting purposes
  dplyr::mutate(treat = factor(ifelse(treat == 1, "Treated", "Control"))) #labels?
p <- ggplot2::ggplot(plot_df, ggplot2::aes(x = p_logit, y = re78)) +
  ggplot2::facet_grid(~treat) + #plot 2 graphs for treat and control
  ggplot2::geom_point() + #scatterplot
  ggplot2::scale_y_continuous(breaks=seq(0,20000,by=1000)) + #?
  ggplot2::geom_smooth(method = "loess", formula = y ~ x, span = 2,
                                      method.args = list(degree = 1)) +
  #what is span of 0.5? is it of x?
  ggplot2::ylab("Real Earnings in 1978") +
  ggplot2::xlab("Estimated Propensity Score") +
  ggplot2::labs(caption = "Data Source: NSW-PSID1.") + ggplot2::theme_bw()

p
```



Data Source: NSW-PSID1.

```
# Save plot object to PDF file
# ggplot2::ggsave(filename_plot_2, plot = p)
# the save function is not working: it says cannot find target with the name "filename_plot_
2"
```

In this changed code, I set the span to “2”. I think the span means diameter of the neighborhood around a chosen point. So intuitively, setting span to 2 should regress all the points in each groups, i.e. linear regression.

Indeed the fitted curve is a line.

But question: what are the chosen points? How are they chosen? Are every point chosen and then each “small” regression lines are “stitched” together?

Q2 b The graphs (and regression lines) can give us a very good feeling of CATE for a given p-score. ATE, which is the integral of CATE based on the distribution of p_score? Maybe not so much.

However, I guess that it seem that the entire data set is so “clumped up” around $\hat{p} = 0$ (since there are a lot more untreated than treated in the sample) that CATE for $\hat{p} = 0$ should be pretty telling of ATE.

CATE: $\hat{E}[y_{1i} - y_{0i}|p = 0] \approx -2000$. Hmmmm, this doesn’t feel right…

Q3 a The following code initializes a variable called stratum, using the bincode function

```
#First, set the bin bounds
b <- c(0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1)

# Calling .bincode() function
psid_trimmed <- psid_trimmed %>%
  dplyr::mutate(.data = psid_trimmed,
                stratum = .bincode(p_logit, b, TRUE, TRUE))

# Now we gives a summary, grouped by variable stratum:
psid_trimmed %>%
    dplyr::group_by(stratum) %>%
    dplyr::summarize(
      'count of treated units' = sum(treat == 1),
      'count of control units' = sum(treat == 0),
      'max p-score' = max(p_logit),
      'min p-score' = min(p_logit)
    )
```

```
## # A tibble: 10 × 5
##    stratum `count of treated units` `count of control units` max p-sco…¹ min p…²
##      <int>                    <int>                     <int>       <dbl>   <dbl>
##  1       1                       11                      1018      0.0993 6.53e-4
##  2       2                        7                        53      0.199  1.00e-1
##  3       3                       11                        24      0.299  2.03e-1
##  4       4                       16                        17      0.389  3.03e-1
##  5       5                        5                         8      0.497  4.03e-1
##  6       6                       15                         6      0.598  5.02e-1
##  7       7                       14                         8      0.692  6.04e-1
##  8       8                        8                         5      0.798  7.11e-1
##  9       9                       13                         0      0.896  8.02e-1
## 10      10                       79                         7      0.974  9.00e-1
## # … with abbreviated variable names ¹`max p-score`, ²`min p-score`
```

Q3 b. We now try to test CIA:

```
# Declare a function that identifies perfectly collinear covariates.
# Note: this fnc is roughly equivalent to _rmcoll in STATA.
rmcoll <- function(df, colnames = names(df)) {
  # Arguments:
  # - df: A data frame.
  # - colnames: A list of column names.
  # Returns:
  # A list with the column names of collinear variables.
    df_ <- df %>% dplyr::select(one_of(colnames))
    cc <- coef(lm(rep(1, nrow(df_)) ~ ., data = df_))
    return(names(cc)[is.na(cc)])
} # end fnc rmcoll

#Listing OPVs
list_OPVs <- c('age', 'edu', 'married', 'black', 'hisp', 're74', 're75', 'u74black')

#Now for each strata 1-10:
for(s in 1:10){
  #set df to this stratum
  df <- psid_trimmed[psid_trimmed$stratum == s]
  #if treated AND control counts positive:
  if(nrow(df[df$treat==1]) >0
     & nrow(df[df$treat==0])>0
  ){
    #get rid of colinear terms
    to_die <- rmcoll(df = df)
    list_survivors <- list_OPVs[!(list_OPVs %in% to_die)]
    #Now get formulas for SUR
    list_fo = list()
    for (x in list_survivors){
      xname = toString(x)
      fo <- as.formula(paste(xname, "~treat"))
      list_fo <- append(list_fo, fo)
    }
    #Now use SUR estimation on this list of formulas
    sur_fit <- systemfit::systemfit(formula = list_fo, data=df, method="SUR")
    #Print summary
    print(paste('stratum:', s))
    print(summary(sur_fit)$coefficients)
    rm(sur_fit)
  }
}
```

```
## [1] "stratum: 1"
##                      Estimate   Std. Error      t value     Pr(>|t|)
## eq1_(Intercept)  3.142731e+01 2.987026e-01 105.21269648 0.000000e+00
## eq1_treat       -1.063672e+00 2.889019e+00  -0.36817761 7.128167e-01
## eq2_(Intercept)  1.122986e+01 8.211127e-02 136.76395722 0.000000e+00
## eq2_treat        3.155921e-01 7.941712e-01   0.39738545 6.911658e-01
## eq3_(Intercept)  8.280943e-01 1.183963e-02  69.94259616 0.000000e+00
## eq3_treat       -9.912484e-03 1.145116e-01  -0.08656316 9.310356e-01
## eq4_(Intercept)  3.624754e-01 1.508133e-02  24.03471984 0.000000e+00
## eq4_treat        2.738882e-01 1.458649e-01   1.87768373 6.070766e-02
## eq5_(Intercept)  4.911591e-02 6.743567e-03   7.28337289 6.474821e-13
## eq5_treat       -4.911591e-02 6.522304e-02  -0.75304547 4.515952e-01
## eq6_(Intercept)  1.221653e+04 2.945914e+02  41.46939437 0.000000e+00
## eq6_treat       -7.390745e+01 2.849256e+03  -0.02593921 9.793109e-01
## eq7_(Intercept)  1.035292e+04 2.598232e+02  39.84603724 0.000000e+00
## eq7_treat       -1.388451e+03 2.512981e+03  -0.55251149 5.807182e-01
## eq8_(Intercept)  1.473477e-02 3.759786e-03   3.91904621 9.478722e-05
## eq8_treat       -1.473477e-02 3.636423e-02  -0.40519963 6.854152e-01
## [1] "stratum: 2"
##                      Estimate   Std. Error    t value     Pr(>|t|)
## eq1_(Intercept)  2.803774e+01 1.095230e+00 25.5998551 0.000000e+00
## eq1_treat        6.765499e-01 3.206506e+00  0.2109929 8.336324e-01
## eq2_(Intercept)  1.043396e+01 3.188134e-01 32.7274907 0.000000e+00
## eq2_treat        2.803235e-01 9.333900e-01  0.3003283 7.650006e-01
## eq3_(Intercept)  5.660377e-01 6.718625e-02  8.4249048 1.193090e-11
## eq3_treat        2.911051e-01 1.967012e-01  1.4799357 1.443020e-01
## eq4_(Intercept)  6.037736e-01 6.842767e-02  8.8235300 2.595701e-12
## eq4_treat       -3.234501e-02 2.003357e-01 -0.1614541 8.722969e-01
## eq5_(Intercept)  7.547170e-02 4.083818e-02  1.8480672 6.969332e-02
## eq5_treat        2.102426e-01 1.195619e-01  1.7584407 8.394786e-02
## eq6_(Intercept)  5.989884e+03 7.517096e+02  7.9683489 6.920553e-11
## eq6_treat       -3.958227e+03 2.200780e+03 -1.7985562 7.729305e-02
## eq7_(Intercept)  3.916448e+03 4.851172e+02  8.0731998 4.617906e-11
## eq7_treat       -6.664367e+02 1.420278e+03 -0.4692298 6.406645e-01
## eq8_(Intercept)  5.660377e-02 3.463412e-02  1.6343355 1.076053e-01
## eq8_treat        8.625337e-02 1.013983e-01  0.8506390 3.984681e-01
## [1] "stratum: 3"
##                      Estimate   Std. Error     t value     Pr(>|t|)
## eq1_(Intercept)  2.783333e+01 1.780308e+00 15.63399537 0.000000e+00
## eq1_treat        7.575758e-02 3.175652e+00  0.02385575 9.811113e-01
## eq2_(Intercept)  1.058333e+01 5.359190e-01 19.74800785 0.000000e+00
## eq2_treat       -4.015152e-01 9.559537e-01 -0.42001527 6.771965e-01
## eq3_(Intercept)  2.916667e-01 9.125041e-02  3.19633262 3.062546e-03
## eq3_treat       -1.098485e-01 1.627693e-01 -0.67487223 5.044582e-01
## eq4_(Intercept)  6.666667e-01 1.008838e-01  6.60826550 1.628960e-07
## eq4_treat       -1.212121e-01 1.799529e-01 -0.67357679 5.052711e-01
## eq5_(Intercept)  8.333333e-02 4.811252e-02  1.73205081 9.260369e-02
## eq5_treat       -8.333333e-02 8.582144e-02 -0.97100831 3.386079e-01
## eq6_(Intercept)  6.476191e+03 1.309017e+03  4.94737086 2.153903e-05
## eq6_treat       -5.914572e+02 2.334978e+03 -0.25330307 8.016054e-01
## eq7_(Intercept)  2.417831e+03 5.528861e+02  4.37310980 1.153742e-04
```

```
## eq7_treat        2.764310e+02 9.862189e+02  0.28029378 7.810018e-01
## eq8_(Intercept)  8.333333e-02 5.884434e-02  1.41616555 1.660974e-01
## eq8_treat        7.575758e-03 1.049645e-01  0.07217448 9.428987e-01
## [1] "stratum: 4"
##                      Estimate   Std. Error     t value     Pr(>|t|)
## eq1_(Intercept)  2.752941e+01 2.190732e+00 12.56630576 1.054712e-13
## eq1_treat        7.205882e-01 3.146200e+00  0.22903449 8.203464e-01
## eq2_(Intercept)  1.064706e+01 4.585677e-01 23.21807321 0.000000e+00
## eq2_treat       -5.845588e-01 6.585677e-01 -0.88762140 3.815797e-01
## eq3_(Intercept)  2.352941e-01 1.072232e-01  2.19443250 3.582339e-02
## eq3_treat        1.470588e-02 1.539876e-01  0.09550042 9.245320e-01
## eq4_(Intercept)  8.823529e-01 8.166561e-02 10.80446212 4.892309e-12
## eq4_treat       -7.352941e-03 1.172833e-01 -0.06269385 9.504128e-01
## eq5_(Intercept)  4.790868e+03 1.301647e+03  3.68061991 8.797585e-04
## eq5_treat        1.042220e+02 1.869348e+03  0.05575315 9.558963e-01
## eq6_(Intercept)  3.134960e+03 6.371173e+02  4.92053776 2.696792e-05
## eq6_treat       -2.293450e+02 9.149901e+02 -0.25065294 8.037363e-01
## eq7_(Intercept)  1.764706e-01 9.650524e-02  1.82861156 7.709417e-02
## eq7_treat        1.102941e-02 1.385951e-01  0.07958010 9.370827e-01
## [1] "stratum: 5"
##                      Estimate   Std. Error       t value     Pr(>|t|)
## eq1_(Intercept)  2.887500e+01 3.329747e+00  8.671830e+00 3.010287e-06
## eq1_treat       -1.075000e+00 5.369056e+00 -2.002214e-01 8.449610e-01
## eq2_(Intercept)  1.062500e+01 8.599353e-01  1.235558e+01 8.611639e-08
## eq2_treat       -1.225000e+00 1.386604e+00 -8.834534e-01 3.958893e-01
## eq3_(Intercept)  2.500000e-01 1.305582e-01  1.914854e+00 8.186423e-02
## eq3_treat       -2.500000e-01 2.105188e-01 -1.187542e+00 2.600246e-01
## eq4_(Intercept)  1.000000e+00 1.167748e-01  8.563488e+00 3.400205e-06
## eq4_treat       -6.000000e-01 1.882938e-01 -3.186510e+00 8.660183e-03
## eq5_(Intercept) -6.427538e-16 9.534626e-02 -6.741259e-15 1.000000e+00
## eq5_treat        2.000000e-01 1.537412e-01  1.300887e+00 2.198878e-01
## eq6_(Intercept)  2.375366e+03 1.799947e+03  1.319687e+00 2.137457e-01
## eq6_treat        3.340514e+03 2.902327e+03  1.150978e+00 2.741463e-01
## eq7_(Intercept)  3.204679e+03 8.124514e+02  3.944456e+00 2.294418e-03
## eq7_treat       -1.287417e+03 1.310039e+03 -9.827321e-01 3.468606e-01
## [1] "stratum: 6"
##                      Estimate   Std. Error    t value     Pr(>|t|)
## eq1_(Intercept)  2.500000e+01    2.9343698  8.5197168 6.499387e-08
## eq1_treat       -1.600000e+00    3.4719932 -0.4608304 6.501515e-01
## eq2_(Intercept)  9.333333e+00    0.7738413 12.0610427 2.377996e-10
## eq2_treat        7.333333e-01    0.9156214  0.8009133 4.330837e-01
## eq3_(Intercept)  1.666667e-01    0.1244873  1.3388252 1.964272e-01
## eq3_treat       -1.000000e-01    0.1472953 -0.6789083 5.053825e-01
## eq4_(Intercept)  8.333333e-01    0.1972027  4.2257713 4.576102e-04
## eq4_treat       -2.333333e-01    0.2333333 -1.0000000 3.298768e-01
## eq5_(Intercept)  1.666667e-01    0.1500487  1.1107503 2.805353e-01
## eq5_treat       -3.333333e-02    0.1775400 -0.1877511 8.530624e-01
## eq6_(Intercept)  3.173687e+03 1213.1380722  2.6160969 1.699093e-02
## eq6_treat       -7.707301e+02 1435.4043246 -0.5369428 5.975394e-01
## eq7_(Intercept)  2.878540e+03  963.2564393  2.9883423 7.554135e-03
## eq7_treat       -1.147385e+03 1139.7403893 -1.0067070 3.267252e-01
```

```
## eq8_(Intercept)  1.666667e-01    0.1244873  1.3388252 1.964272e-01
## eq8_treat       -1.000000e-01    0.1472953 -0.6789083 5.053825e-01
## [1] "stratum: 7"
##                       Estimate   Std. Error    t value     Pr(>|t|)
## eq1_(Intercept)   24.87500000    2.4626903 10.1007421 2.672311e-09
## eq1_treat         -0.37500000    3.0871457 -0.1214714 9.045298e-01
## eq2_(Intercept)    9.87500000    0.8580631 11.5084773 2.836402e-10
## eq2_treat          1.19642857    1.0756390  1.1122956 2.792028e-01
## eq3_(Intercept)    0.12500000    0.1061712  1.1773439 2.528796e-01
## eq3_treat         -0.05357143    0.1330926 -0.4025124 6.915767e-01
## eq4_(Intercept)    0.87500000    0.1061712  8.2414072 7.341425e-08
## eq4_treat          0.05357143    0.1330926  0.4025124 6.915767e-01
## eq5_(Intercept) 1533.12586212  803.3937028  1.9083120 7.080955e-02
## eq5_treat        736.04129573 1007.1073044  0.7308469 4.733485e-01
## eq6_(Intercept)  643.84411430  446.0506517  1.4434327 1.643803e-01
## eq6_treat        586.66101810  559.1540834  1.0491938 3.066027e-01
## eq7_(Intercept)    0.12500000    0.1421298  0.8794775 3.895875e-01
## eq7_treat          0.08928571    0.1781692  0.5011288 6.217514e-01
## [1] "stratum: 8"
##                   Estimate   Std. Error    t value     Pr(>|t|)
## eq1_(Intercept)    24.800    2.9007836  8.5494140 3.454730e-06
## eq1_treat           7.200    3.6977880  1.9471100 7.749806e-02
## eq2_(Intercept)    10.800    0.3384456 31.9105894 3.404610e-12
## eq2_treat           0.450    0.4314352  1.0430302 3.193155e-01
## eq3_(Intercept)     0.200    0.2205365  0.9068792 3.839067e-01
## eq3_treat           0.425    0.2811300  1.5117560 1.587800e-01
## eq4_(Intercept)     0.800    0.1206045  6.6332496 3.693776e-05
## eq4_treat           0.200    0.1537412  1.3008873 2.198878e-01
## eq5_(Intercept)  1567.414  416.0428890  3.7674337 3.114223e-03
## eq5_treat       -1567.414  530.3527024 -2.9554182 1.308203e-02
## eq6_(Intercept)  2539.034 1509.6643619  1.6818531 1.207349e-01
## eq6_treat       -1037.106 1924.4520101 -0.5389099 6.006855e-01
## [1] "stratum: 10"
##                      Estimate   Std. Error     t value  Pr(>|t|)
## eq1_(Intercept)  23.28571429   1.86344307 12.49606960 0.0000000
## eq1_treat         0.08137432   1.94424866  0.04185387 0.9667145
## eq2_(Intercept)  10.57142857   0.69323142 15.24949423 0.0000000
## eq2_treat        -0.40687161   0.72329243 -0.56252712 0.5752551
## eq3_(Intercept)   1.00000000   0.08036342 12.44347218 0.0000000
## eq3_treat        -0.05063291   0.08384827 -0.60386353 0.5475617
## eq4_(Intercept) 127.88014439 284.84597799  0.44894488 0.6546275
## eq4_treat       106.59136339 297.19792410  0.35865447 0.7207534
```

We have now tested balance in non-colinear OPV's in each of the strata.

Why is this a good idea? Even if OPV's are balanced across treated and control groups in the larger sample, we have now subdivided the sample based on p-scores. This means that now each bins are an even smaller samples. Even if our strata are assigned randomly, the probability of OPVs not balanced just by random chance increases. We even see that some strata do not even have any controls/treated individuals in them.

As for findings, it seems that in each strata, the coefficients on the treatments do not generally have low p-values. This suggests that in general, the OPVs are balanced across treatment and control groups in each

strata.

Q3 c

We implement the Stratification Matching Estimator of ATT here

```
#First, we assign the weights and difference in CATT (?) in one loop
w = list()
d = list()
for(s in 1:10){
  #set df to this stratum
  df <- psid_trimmed[psid_trimmed$stratum == s]
  #if treated AND control counts positive:
  if(nrow(df[df$treat==1]) >0
     & nrow(df[df$treat==0])>0
  ){
    #num of treated in each strata
    n <- nrow(df[df$treat == 1])
    w <- c(w, n)

    #Find avg treatment effect in stratum
    avg_t <- mean(df[df$treat == 1]$re78)
    avg_c <- mean(df[df$treat == 0]$re78)
    diff <- avg_t-avg_c
    d <- append(d, c(diff))
  }
}

#make appended list "normal"
d <- unlist(d)
w <- unlist(w)

#normalize weight
w <- w/sum(w)


#Now for the estimator:
att <- weighted.mean(d, w)
att
```

```
## [1] 1562.727
```

Indeed, we get (approximately) the right amount!

Now why is this the estimator of ATT? We know that for the treated individuals, their realized potential outcome is their incomes in ’78 after receiving treatment. But how do we know about their incomes in a world in which they never got the treatment? We need an estimator for that.

Our estimator is the average income of people similar to the treatment (by pre-treatment OPVs), but who did not receive the treatment, i.e. the control group in each strata. We defined “similar” by strata of p-score. Since

each strata features people with “close enough” p-scores, we think that they should be similar in terms of pre-treatment OPVs.

But how is it that avg. income of the control individuals in each strata is a “reasonable” estimator of the avg. income of the treated individuals had they not received treatment? We have shown in Q3 that within each stratum, whether an individual received treatment does not predict any of the pre-treatment OPVs. This suggests that within each stratum, treatment assignment is unlikely to be correlated with the outcomes of individuals if everyone is untreated. Hence, the avg. income of the control individuals is an unbiased estimator of avg. income of treated individuals had they not been treated.

Using this estimator, and the treated count weights of each strata as estimator for the probability distribution of treated individuals across different p-scores, we now have an estimator for avg. treatment effect on the treated, ATT.

Part ii

Q4

What does unconditional RA do to p-score of individuals with any x, vector of realized OPVs? It sets them the same:

$$\forall x,\, x' \in X\; \hat{p}(x) \equiv Pr[D=1|x] = \hat{p}(x')$$

(D is treatment assignment, as in class). Let this constant p-score be p.

Now if p-scores across x are all the same, what does that mean for the 3 different kinds of matching correspondence? They yield the same result: What does unconditional RA do to p-score of individuals with any x, vector of realized OPVs? It sets them the same:

$$\foralli\; C^{NNM}(i) = C^{RM}(i) = C^{KM}(i) = C$$

Since everyone has the same p-score, they are all nearest neighbors and within radius $r > 0$ of each other.

It also follows that the weight functions are the same. Cares need to be taken for KM weights though:

$$\begin{aligned}\forall i,j\; K(\frac{\hat{p}_i - \hat{p}_j}{h}) &= K(0)\\ \Rightarrow w_{ij} &= K(0)/\sum_C K(0) = 1/N^C\end{aligned}$$

This means that the estimators from NNM, RM, and KM are the same:

$$\begin{aligned}\widehat{ATT}^m &= \frac{1}{N^T}\sum_{i\in T}\left[y_i - \sum_{j\in C(i)}\frac{1}{N^C}y_j\right], \forall m \in \{NNM, RM, KM\}\\ &= \frac{1}{N^T}\sum_{i\in T} y_i - \frac{1}{N^T}\sum_{i\in T}\bar{y}^0\\ &= \bar{y}^1 - \bar{y}^0\end{aligned}$$

Q5

$$\begin{aligned}\widehat{ATT}^m &= \frac{1}{N^T}\sum_{i\in T}\left[y_i - \sum_{j\in C^m(i)} w_{ij}^m y_j\right], m \in \{NNM, RM, KM\} \\ &= \frac{1}{N^T}\sum_{i\in T} y_i - \frac{1}{N^T}\sum_{i\in T}\sum_{j\in C} w_{ij} y_j \\ &= \bar{y}^1 - \frac{1}{N^C}\sum_{j\in C}(\frac{N^C}{N^T}\sum_{i\in T} w_{ij}) y_j\end{aligned}$$

Hence $\pi_j = \frac{N^T}{N^C}\sum_{i\in T} w_{ij}$.

Q6

We now present the 1-1 NMM estimator of ATT:

```
X <- psid_trimmed$p_logit
Tr<- psid_trimmed$treat
Y <- psid_trimmed$re78

install_packages_if_needed(c("Matching"))

rr  <- Matching::Match(Y=Y, Tr=Tr, X=X, M=1, estimand='ATT');

summary(rr)
```

```
##
## Estimate...  490.39
## AI SE......  1929.6
## T-stat.....  0.25414
## p.val......  0.79939
##
## Original number of observations..............  1325
## Original number of treated obs...............  179
## Matched number of observations...............  179
## Matched number of observations  (unweighted).  700
```

This doesn't instill confidence. Why is it so different to Naive estimate?

Q7

First, we create the data frame to store the pairings and related info:

```
dt_pairing <- data.table::data.table(
  index_treated = rr$index.treated,
  index_control = rr$index.control,
  p_treated = psid_trimmed[rr$index.treated]$p_logit,
  p_control = psid_trimmed[rr$index.control]$p_logit,
  y_treated = psid_trimmed[rr$index.treated]$re78,
  y_control = psid_trimmed[rr$index.control]$re78,
  weight = rr$weights
)
# dt_pairing <-dt_pairing %>%
#   dplyr::mutate(.data=dt_pairing,
#
#                 )
```

a

We count the number of unique treated indices in dt_pairing:

```
length(unique(dt_pairing$index_treated))
```

```
## [1] 179
```

There are 179 treated individuals paired (i.e. all of the treated).

b

Here we count the pairs with weight 1:

```
nrow(dt_pairing[dt_pairing$weight == 1])
```

```
## [1] 151
```

There are 151 treated individuals paired with exactly 1 control unit.

c

Here we find the percentage of unique control units to the total number of control units:

```
length(unique(dt_pairing$index_control))/nrow(psid_trimmed[psid_trimmed$treat == 0, ])
```

```
## [1] 0.4668412
```

Only 46.7 percent of control units are paired.

d & e

```
length(unique(dt_pairing[dt_pairing$weight == 1]$index_control))
```

```
## [1] 34
```

Only 34 distinct control units are used to pair up with the 151 treated units with only one pairing.

What is going on? This also serves as answer to (e): Since we see in the scatter plot that control and treatment groups are very clustered in p-score, when we need to find their closest neighbors, we would often land on the rare instances of either case.

For an analogy, since we are in the world of 1-to-1 pairing: dating scene! There are a lot more boys in the Math department than girls, so when you ask boys who he likes the most (the “pairings”), their answers can only be so varied. On the flip side, the gender imbalance is reversed in the Art department, so the opposite happens.

This problem would not be as egregious if the control and treatment groups are both evenly distributed in p-score, but by definition of p-score (conditional probability of assignment given x) and MLE, that is not gonna happen, is it?

f

Is this just the number of matchings? Which is $\sum_i N_i^C$?

g

```
d <- mapply('-', dt_pairing$y_treated, dt_pairing$y_control, SIMPLIFY = FALSE)
sum(unlist(d)*dt_pairing$weight)/nrow(psid_trimmed[psid_trimmed$treat ==1])
```

```
## [1] 490.3947
```

We get the right answer by hand.

h

```
rr2  <- Matching::Match(Y=Y, Tr=Tr, X=X, M=1, estimand='ATT', ties = FALSE);

summary(rr2)
```

```
##
## Estimate...  613.68
## SE.........  811.51
## T-stat.....  0.75623
## p.val......  0.44951
##
## Original number of observations..............  1325
## Original number of treated obs...............  179
## Matched number of observations...............  179
## Matched number of observations  (unweighted).  179
```

In this case, we implement a tie breaker in finding matches. Hence, each treatment unit only has one control unit matched with it.

Does this improve NNM performance? I don't know. It doesn't seem to be the case.

Q8

Here we find the balance object:

```
nnm_balance <- Matching::MatchBalance(treat~age + edu + black +
                    hisp + married + nodegree + re74 + re75 +
                    u74 + u75, data=psid_trimmed, match.out=rr, nboots=500)
```

```
## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似

## Warning in ks.test.default(...): 并列的时候P-值将近似
```

```
##
## ***** (V1) age *****
##                         Before Matching        After Matching
## mean treatment........     25.765             25.765
## mean control..........     30.962             26.172
## std mean diff.........     -71.52            -5.6024
##
## mean raw eQQ diff.....     5.2011             4.5157
## med  raw eQQ diff.....          4                  3
## max  raw eQQ diff.....         11                 17
##
## mean eCDF diff........    0.13324            0.11579
## med  eCDF diff........    0.13914            0.11286
## max  eCDF diff........    0.22992            0.30286
##
## var ratio (Tr/Co).....    0.58984             1.1862
## T-test p-value........ 1.1102e-15             0.5144
## KS Bootstrap p-value.. < 2.22e-16         < 2.22e-16
## KS Naive p-value...... 1.5576e-07         < 2.22e-16
## KS Statistic..........    0.22992            0.30286
##
##
## ***** (V2) edu *****
##                         Before Matching        After Matching
## mean treatment........     10.413             10.413
## mean control..........     11.141             10.329
## std mean diff.........     -36.26             4.2181
##
## mean raw eQQ diff.....    0.91061            0.85143
## med  raw eQQ diff.....          1                  1
## max  raw eQQ diff.....          3                  3
##
## mean eCDF diff........   0.056089           0.056667
## med  eCDF diff........   0.029373           0.052857
## max  eCDF diff........    0.28646            0.33286
##
## var ratio (Tr/Co).....    0.60002            0.95434
## T-test p-value........ 2.1575e-05            0.68046
## KS Bootstrap p-value.. < 2.22e-16         < 2.22e-16
## KS Naive p-value...... 1.8471e-11         < 2.22e-16
## KS Statistic..........    0.28646            0.33286
##
##
## ***** (V3) black *****
##                         Before Matching        After Matching
## mean treatment........    0.83799            0.83799
## mean control..........    0.40401            0.91259
## std mean diff.........     117.45            -20.189
##
## mean raw eQQ diff.....    0.43575            0.42571
## med  raw eQQ diff.....          0                  0
```

```
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........    0.21699           0.21286
## med  eCDF diff........    0.21699           0.21286
## max  eCDF diff........    0.43397           0.42571
##
## var ratio (Tr/Co).....    0.56651            1.7019
## T-test p-value........ < 2.22e-16          0.024951
##
##
## ***** (V4) hisp *****
##                          Before Matching        After Matching
## mean treatment........   0.061453          0.061453
## mean control..........   0.051483         0.0052159
## std mean diff.........     4.1394            23.351
##
## mean raw eQQ diff.....  0.0055866              0.02
## med  raw eQQ diff.....          0                 0
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........  0.0049845              0.01
## med  eCDF diff........  0.0049845              0.01
## max  eCDF diff........  0.0099691              0.02
##
## var ratio (Tr/Co).....     1.1867            11.116
## T-test p-value........    0.60314         0.0032282
##
##
## ***** (V5) married *****
##                          Before Matching        After Matching
## mean treatment........    0.19553           0.19553
## mean control..........    0.77574            0.1718
## std mean diff.........    -145.88             5.967
##
## mean raw eQQ diff.....    0.58101              0.08
## med  raw eQQ diff.....          1                 0
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........    0.29011              0.04
## med  eCDF diff........    0.29011              0.04
## max  eCDF diff........    0.58021              0.08
##
## var ratio (Tr/Co).....    0.90847            1.1055
## T-test p-value........ < 2.22e-16          0.49461
##
##
## ***** (V6) nodegree *****
##                          Before Matching        After Matching
## mean treatment........    0.69832           0.69832
## mean control..........    0.41187            0.5472
## std mean diff.........     62.236            32.834
```

```
##
## mean raw eQQ diff.....    0.28492           0.33286
## med  raw eQQ diff.....          0                 0
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........    0.14323           0.16643
## med  eCDF diff........    0.14323           0.16643
## max  eCDF diff........    0.28646           0.33286
##
## var ratio (Tr/Co).....    0.87381           0.85025
## T-test p-value........ 4.0146e-13        0.00081326
##
##
## ***** (V7) re74 *****
##                         Before Matching        After Matching
## mean treatment........     2165.8             2165.8
## mean control..........      11386             1811.3
## std mean diff.........    -186.17             7.1571
##
## mean raw eQQ diff.....     9620.4             3445.1
## med  raw eQQ diff.....      11201             2157.9
## max  raw eQQ diff.....     102109              13527
##
## mean eCDF diff........    0.37508           0.11119
## med  eCDF diff........    0.43034          0.094286
## max  eCDF diff........    0.61128           0.30143
##
## var ratio (Tr/Co).....    0.28201             1.7951
## T-test p-value........ < 2.22e-16           0.27309
## KS Bootstrap p-value.. < 2.22e-16        < 2.22e-16
## KS Naive p-value...... < 2.22e-16        < 2.22e-16
## KS Statistic..........    0.61128           0.30143
##
##
## ***** (V8) re75 *****
##                         Before Matching        After Matching
## mean treatment........     1583.4             1583.4
## mean control..........     9528.6             1506.1
## std mean diff.........    -243.68             2.3703
##
## mean raw eQQ diff.....     8555.8             4593.2
## med  raw eQQ diff.....     9452.9             4662.5
## max  raw eQQ diff.....     131511              10820
##
## mean eCDF diff........     0.3912            0.15447
## med  eCDF diff........    0.45985           0.13571
## max  eCDF diff........    0.59525           0.37286
##
## var ratio (Tr/Co).....    0.15723             1.1196
## T-test p-value........ < 2.22e-16           0.73692
## KS Bootstrap p-value.. < 2.22e-16        < 2.22e-16
```

```
## KS Naive p-value..... < 2.22e-16        < 2.22e-16
## KS Statistic.........    0.59525           0.37286
##
##
## ***** (V9) u74 *****
##                          Before Matching        After Matching
## mean treatment........    0.69832           0.69832
## mean control..........    0.15707            0.6214
## std mean diff.........     117.59            16.713
##
## mean raw eQQ diff.....     0.5419              0.04
## med  raw eQQ diff.....          1                 0
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........    0.27063              0.02
## med  eCDF diff........    0.27063              0.02
## max  eCDF diff........    0.54126              0.04
##
## var ratio (Tr/Co).....     1.5987           0.89546
## T-test p-value........ < 2.22e-16          0.048572
##
##
## ***** (V10) u75 *****
##                          Before Matching        After Matching
## mean treatment........    0.58659           0.58659
## mean control..........    0.20332           0.58163
## std mean diff.........     77.614            1.0056
##
## mean raw eQQ diff.....    0.37989          0.087143
## med  raw eQQ diff.....          0                 0
## max  raw eQQ diff.....          1                 1
##
## mean eCDF diff........    0.19164          0.043571
## med  eCDF diff........    0.19164          0.043571
## max  eCDF diff........    0.38328          0.087143
##
## var ratio (Tr/Co).....     1.5042            0.99657
## T-test p-value........ < 2.22e-16             0.9165
##
##
## Before Matching Minimum p.value: < 2.22e-16
## Variable Name(s): age edu black married re74 re75 u74 u75  Number(s): 1 2 3 5 7 8 9 10
##
## After Matching Minimum p.value: < 2.22e-16
## Variable Name(s): age edu re74 re75  Number(s): 1 2 7 8
```

It gave some sort of warning on p-score (?)

The covariates seem to be more balanced after matching: the differences between avg. in control and in treatment all shrinks. But how de we evaluate the performance of matching estimator in balancing the sample vs, say naive stratification? Comparing the p-value for age prediction, for instance. It seems that naive

stratification has higher p-value in each strata than the matching differences. I am not sure if this is evidence enough, but I suspect that matching may perform worse by pairing up distant units while stratification would not (instead having 0 control/treatment in some strata, a different kind of imbalance).

Q9

What are the takeaways? Perhaps most outstanding issue is the divergence in estimates using different matching estimators. Mechanically, different matching estimator would lead to diverging results depending on the distributions of p-scores within treatment and control groups. So care should be taken to examine what the distributions of p-scores look like, and what estimators would face troubles with it.

A second issue is trimming the sample. Our common support is very generous in this case: the lower bound is very close to 0, and the upper bound very close to 1. While it trimmed about half of the original sample, matching estimator shows us that there is still a lot of clustering around 0 and 1 in p-score based on treatment status. Hence we may consider more aggressive trimming methods. Although this may push samples to be too small, weakening explanatory and predictive power of our results.