

Lab 4, modified 2/27/09; see also Rogosa R-session

Stat 209 Lab: Matched Sets in R
Lab prepared by Karen Kapur.

1 Motivation

1. Suppose we are trying to measure the effect of a treatment variable on the response. In observational studies unbalanced covariates can affect the assignment of treatment. For example, in the ensuing analysis we are trying to determine how further job training affects salaries. However, it may be the case that the people who seek additional job training tend to be more motivated and as a result will tend to have higher salaries. Such a confounding variable would obscure the analysis.
2. One solution is to construct control and treated groups with similar distributions of covariates \mathbf{x} . We can create matched sets within which we can analyze the treatment effects.

2 Description of the Data

1. Begin by installing the packages MatchIt, ~~optmatch~~ ~~Zelig~~ if they are not already installed. Access the packages with the library command.

```
library(MatchIt)
library(optmatch)
library(Zelig)
data(lalonde)
```

2. The data consists of 10 variables measured for each individual, an indicator of treatment assignment, age in years, education in years, an indicator for African-American, black, an indicator for Hispanic, hispan, an indicator for married, married, an indicator for high school degree, nodegree, income in 1974, re74, income in 1975 re75, and income in 1978, re78.

```
lalonde[1:10,]
```

3 Illustrating Example

1. As the number of covariates increases it becomes increasingly difficult to match treatments and controls with the same or similar values of \mathbf{x} . For illustration, take the first observation. Select the number of individuals with the same values of covariates (age within 6 years and education within 6 years, ignoring variables re74 and re75). Exercise: How would you modify this to select individuals with re74 and re75 within \$ 6000?

```
numObs <- dim(lalonde)[1]
indiv1 <- lalonde[1,]
matches <- matrix( nrow = numObs, ncol = 1, data = FALSE )
for ( i in 1:numObs) {
  matches[i,1] <- all(
    c(lalonde[i,4:7] == indiv1[4:7],
      (lalonde[i,3] < indiv1[3] + 3) && (lalonde[i,3] > indiv1[3] - 3),
      (lalonde[i,2] < indiv1[2] + 3) && (lalonde[i,2] > indiv1[2] - 3)
    ) )
}
sum(as.numeric(matches) )
```

2. The propensity score is defined as the probability of receiving the treatment given the observed covariates \mathbf{x} . These scores are used to construct matched sets or strata. We construct propensity scores by fitting a logistic regression to the data.

```
glm.lalonde <- glm( treat ~ age + educ + black +
  hispan + married + nodegree + re74 + re75,
  data = lalonde, family = binomial)
summary(glm.lalonde)
propScores <- glm.lalonde$fitted
```

3. We will split up the cases into 5 groups, defined by the quintiles of the propensity scores. Within each quintile, we can analyze the effect of treatment on the response.

```
quintilePropScores <- quantile( propScores,
  probs = seq( from = 0, to = 1, by = .2) )
lalonde$PropScores <- propScores
```

```

treated <- list(5)
controls <- list(5)
for ( i in 1:5 ) {
  treated[[i]] <- subset( x = lalonde,
    subset = PropScores >= quintilePropScores[i] &
    PropScores < quintilePropScores[i+1] &
    treat == 1)
  controls[[i]] <- subset( x = lalonde,
    subset = PropScores >= quintilePropScores[i] &
    PropScores < quintilePropScores[i+1] &
    treat == 0)
}

```

4. Within each group the cases are balanced in the sense that the propensity scores tend to be similar. Therefore, we can directly analyze the effect of treatment on the response. For each group we can perform a t-test to analyze the effect of treatment. I choose a one-sided alternative because we believe that job training will cause income to increase.

```

for ( i in 1:5 ) {
  t.test( treated[[i]]$re78,
    controls[[i]]$re78,
    alternative = "greater" )
}

```

4 Matching via MatchIt

1. More sophisticated matching models can be fit using the MatchIt library. $1:k$ matching is achieved by calling `matchit` with `method = "nearest"`. The default value for k is 1, but it is set here explicitly using the `ratio` argument.

```

m.out <- matchit(treat ~ re74 + re75 + educ + black + hispan + age,
  data = lalonde, method = "nearest", ratio = 1)

```

2. The $1:k$ matching uses the greedy algorithm by default. To use optimal matching, we select `method = "optimal"`.

```

m.out <- matchit(treat ~ re74 + re75 + age + educ, data = lalonde,
  method = "optimal", ratio = 2)

```

3. Full matching is achieved by setting `method = "full"`.

```
m.out <- matchit(treat ~ age + educ + black + hispan + married +
  nodegree + re74 + re75, data = lalonde, method = "full")
```

4. The `summary` command gives statistics to measure the balance between treated and control samples.

```
summary(m.out)
```

5. Interpreting summary statistics:

- Means of each covariate among treated and untreated cases.
- Median, mean, and maximum distance between empirical quantile functions. Values greater than 0 indicate deviations between the groups in some part of the empirical distributions.

6. We can visualize the balance criteria using the `plot` function.

```
plot(m.out, type = "jitter" )
plot(m.out)
```

optional: I decided not to mess with

5 Analyzing the Response

Zelig, which has a lot of overhead. Instead my R-session does a number of descriptive exercises for the matched data

1. We use `match.data()` to create the matched data from the MatchIt output object (`m.out`). The data excludes unmatched units and includes information about the matching (weights, subclasses, and the distance measure).

```
m.data <- match.data(m.out)
```

2. We analyze the matched data using the `zelig` function. First we fit a model using all the matched data.

```
library(Zelig)
z.out0 <- zelig(re78 ~ treat + age + educ + black + hispan + nodegree +
  married + re74 + re75, data = match.data(m.out0),
  model = "ls")
```

3. We can predict the response for controls by setting the explanatory variables at their means (the default). Similarly, we can predict the response for the treatments.

```
x.out0 <- setx(z.out0, treat=0)
x.out1 <- setx(z.out0, treat=1)
```

4. Finally compute the result and examine a summary:

```
s.out0 <- sim(z.out0, x = x.out0)
s.out1 <- sim(z.out0, x = x.out1)
summary(s.out0)
summary(s.out1)
```

5. Next we can fit a model using only the controls. Then we can predict the response for the treatments.

```
z.out1 <- zelig(re78 ~ age + educ + black + hispan + nodegree +
married + re74 + re75, data = match.data(m.out0, "control"),
model = "ls")
x.out1 <- setx(z.out1, data = match.data(m.out1, "treat"), cond = TRUE)
s.out1 <- sim(z.out1, x = x.out1)
summary(s.out1)
```

6. Now we fit a model using only the treated cases. Then we can predict the response for the controls.

```
z.out2 <- zelig(re78 ~ age + educ + black + hispan + nodegree +
married + re74 + re75, data = match.data(m.out0, "treat"),
model = "ls")
x.out2 <- setx(z.out2, data = match.data(m.out1, "control"), cond = TRUE)
s.out2 <- sim(z.out2, x = x.out2)
summary(s.out2)
```

7. We can compare the two treatment effects as follows.

```
ate.all <- c(s.out1$qi$att.ev, -s.out2$qi$att.ev)
mean(ate.all)
sd(ate.all)
quantile(ate.all, c(0.025, 0.975))
```

8. Next we can compare treatment effects within subclasses. In subclassification, the average treatment effect estimates are obtained separately for each subclass, and then aggregated for an overall estimate. Estimating the treatment effects separately for each subclass, and then aggregating across subclasses, can increase the robustness of the ultimate results since the parametric analysis within each subclass requires only local rather than global assumptions. However, fewer observations are obviously available within each subclass, and so this option is normally chosen for larger data sets.

```
m.out2 <- matchit(treat ~ age + educ + black + hispan + nodegree +
  married + re74 + re75, data = lalonde,
  method = "subclass", subclass = 4)
```

9. We regress using the controls and impute the counterfactual outcomes for treated effects.

```
z.out3 <- zelig(re78 ~ re74 + re75 + distance,
  data = match.data(m.out2, "control"),
  model = "ls", by = "subclass")
x.out3 <- setx(z.out3, data = match.data(m.out2, "treat"), fn = NULL,
  cond = TRUE)
s.out3 <- sim(z.out3, x = x.out3)
summary(s.out3)
```

10. It is also possible to get the summary result for each subclass. For example, the following command summarizes the result for the second subclass.

```
summary(s.out3, subset = 2)
```

6 Distance Measures for Matching

1. Matching is done based on propensity scores by default. Another option is to match individuals based on the Mahalanobis distance.

```
matchit.mahal <- matchit( treat ~ age + educ +
  black + hispan + married + nodegree +
  re74 + re75, method = "full",
  distance = "mahalanobis", data = lalonde )
```

2. Exercise. How does the distance method change the analysis of the ~~xxxxxxxxxxxx~~ we did "full" here, how good a match ?

7 References

1. Ho, D.E., Imai, K., King, G., and Stuart, E.A. (2006) MatchIt: Non-parametric Preprocessing for Parametric Causal Inference
2. Rosenbaum, P.R. (2002) Observational Studies.