# 5

# Propensity Score Matching

## Learning Objectives

- Describe and compare greedy, genetic, and optimal matching algorithms.
- Characterize the impact of matching with or without replacement on results and analysis choices.
- Compare one-to-one, fixed ratio, variable ratio, and full matching strategies.
- Implement methods to estimate treatment effects with samples obtained with different matching methods.
- Implement methods to estimate standard errors of treatment effects with samples obtained with different matching methods.
- Understand the rationale and implementation of Rosenbaum's sensitivity analysis.

## 5.1. Introduction

This chapter presents the implementation of different propensity score matching methods, as well as a comparison of methods in terms of covariate balance and bias of treatment effect estimates. Propensity score matching consists of grouping observations with similar values of propensity scores. However, while propensity score weighting (see Chapter 3) and propensity score stratification (see Chapter 4) preserve the original sample size if there is adequate common support, most forms of matching result in the discarding of some observations. The sample size after matching is smallest with one-to-one matching and can vary considerably with variable ratio and full matching strategies. Propensity score matching methods differ in the ratio of treated observations matched to untreated observations, the algorithm used for identifying matches,

whether matches are done with or without replacement, and whether matches are based solely on propensity scores or also use values of covariates. This chapter presents an overview of variations of propensity score matching and demonstrates them with an example. Issues specific to matching methods related to the enforcement of common support, covariate balance evaluation, the estimation of treatment effects, and standard errors are also discussed.

## 5.2. Description of Example

The example for this chapter consists of the estimation of the effect of mothers having a job that provides or subsidizes child care on the length that they breastfeed their children, using data from the National Longitudinal Survey of Youth 1979 (NLSY79) and the NLSY79 Children and Youth. The health and cognitive benefits of breastfeeding on children are well documented (Borra, Iacovou, & Sevilla, 2012; Quigley et al., 2012). Therefore, it is important to understand the factors that lead mothers to initiate and maintain breastfeeding, including job characteristics, because job demands can conflict with breastfeeding efforts. Jacknowitz (2008) examined the effects of mothers having a job that provides or subsidizes child care on whether mothers initiated breastfeeding and on whether they breastfed until the child was 6 months old. Using multiple regression models, she found that mothers who worked for a company that offered child care were more likely to breastfeed to 6 months.

The sample for this example contains 1,209 child records from the NLSY79 Children and Youth data set. The sample is restricted to one child per mother, mothers who had at least one job in the fourth quarter of pregnancy, and mothers who returned to work within 12 weeks of the birth of the child. The NLSY79 data used were restricted to years 1988 to 1994, 1996, 1998, 2000, 2002, 2004, 2006, 2008, and 2010 because responses about the treatment of interest were available for these years. Because the NLSY79 was not designed to be representative of the population of working mothers that is the focus of this example, the NLSY79 sampling weights are not be used in this demonstration (for an example of propensity score analysis with sampling weights, see Chapters 2 and 3).

The outcome variable is the age of the child in weeks when breastfeeding ended. This outcome was taken from the NLSY79 Children and Youth data. The treatment indicator is whether the mother's job provided or subsidized child care and was obtained from the NLSY79. From the data set analyzed, child care was provided or subsidized in 107 (8.85%) of 1,209 cases.

## 5.3. Propensity Score Estimation

In this example, propensity scores for whether the mother's job provided or subsidized child care are estimated using logistic regression. Covariates for the propensity score model selected for this example include variables hypothesized to be true

confounders, because they relate to both the probability of having a job that provides or subsidizes child care and breastfeeding duration (the outcome), as well as predictors of breastfeeding duration, which are included to increase the efficiency of treatment effect estimates. Some examples of covariates are the benefits provided by the mother's current job (i.e., life insurance, dental insurance, profit sharing, retirement, training opportunities), the mother's education level, hours worked per week, and employment sector. The propensity score model also included covariates related to breastfeeding duration controlled by Jacknowitz (2008), such as family size, amount of public assistance received by the family, and whether a cesarean section was performed. A total of 31 covariates were included in the propensity score model. A complete list of covariates used is available in the R code for this chapter in the book's website. A detailed discussion of strategies to select covariates and estimate propensity score models is presented in Chapter 2.

It is advantageous to match on the linear propensity score (i.e., the logit of the propensity score) rather than the propensity score itself, because it avoids compression around 0 and 1 (Diamond & Sekhon, 2013). The linear propensity score is obtained with

$$\log(\mathrm{e}_i(\mathrm{X})) = \log\left(\frac{e_i(X)}{1-e_i(X)}\right), \tag{5.1}$$

where $e_i(X)$ is the estimated propensity score. The following R code shows the use of the *glm* function to fit a logistic regression model to the data,[1] and then linear propensity scores are obtained according to Equation (5.1).

```
psModel = glm(psFormula, data, family=binomial())

data$logitPScores = log(fitted(psModel)/(1-fitted(psModel)))
```

A preliminary evaluation of common support was performed using histograms and box plots of the distributions of linear propensity scores for the treated and untreated. These graphs are shown in Figures 5.1 and 5.2. They clearly indicate that common support is potentially adequate to estimate the ATT with matching methods, because the distribution of the treated is contained within the distribution of the untreated, and therefore an adequate match could be found for every treated observation. However, the use of a caliper during the matching process allows for a more precise evaluation of the adequacy of common support. The graphs also indicate that estimating the ATE using propensity score matching with these data may be difficult because there are areas of the distribution of the untreated without any treated cases nearby, which could result in poor matching. Therefore, the ATT of a mother working for a company that provides or subsidizes child care will be estimated. In applications of propensity score matching, the ATT is more commonly estimated than the ATE.

[1] R code to prepare the data set and to specify the propensity score model in the psFormula object is shown in the book's website.

**FIGURE 5.1 ● Histograms of Linear Propensity Scores for Treated and Untreated Observations**

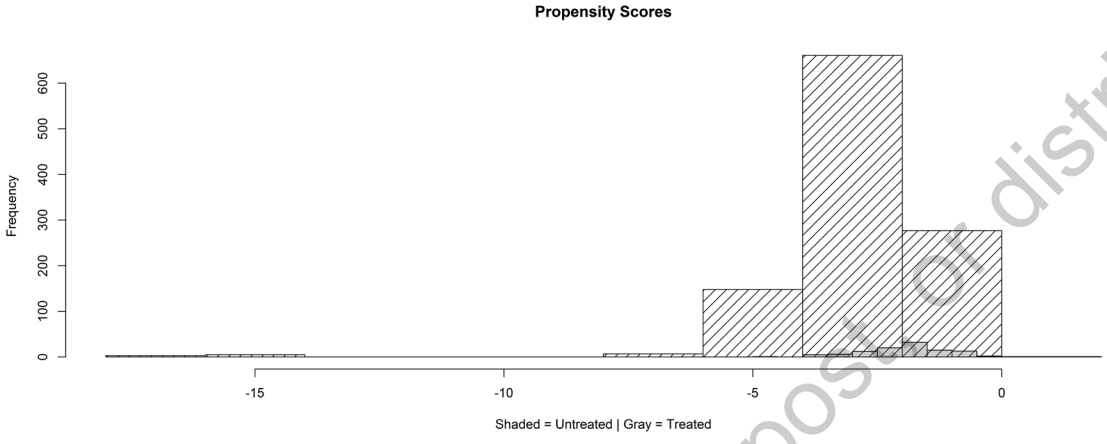
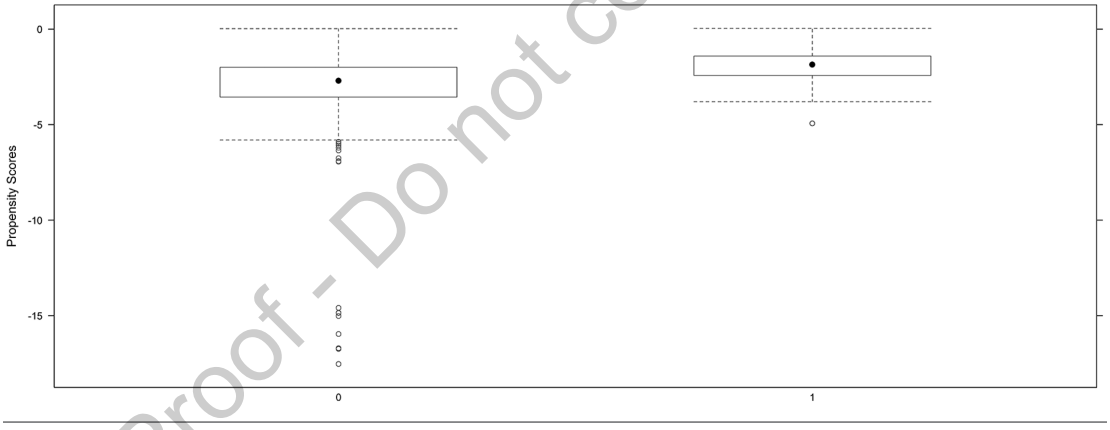



**FIGURE 5.2 ● Box Plots of Linear Propensity Scores for Treated and Untreated Observations**




## 5.4. Propensity Score Matching Algorithm

### 5.4.1. Greedy Matching

Greedy matching consists of choosing each treated case and searching for the best available match among the untreated cases without accounting for the quality of the match of the entire treated sample. Greedy matching contrasts with genetic matching and optimal matching, discussed later in this chapter, which attempt to optimize

global match quality. Greedy matching works well for estimating the ATT when the number of treated cases is substantially smaller than the number of untreated cases available for matching (i.e., the ratio of treated to untreated sample sizes is small) and there is common support for all treated cases. The implementation of greedy matching consists of choosing whether to match with or without replacement, whether to enforce a maximum allowable distance, and whether to allow multiple matching. These options are reviewed below.

***Matching with or without replacement.*** Matching with replacement consists of selecting one or multiple matches for each case (depending on the desired matching ratio) and then returning the matched cases to the pool of observations. In matching without replacement, each case can be used as a match only once. Matching with replacement has the advantage of always matching each treated case to the closest untreated case and therefore produces larger bias reduction than matching without replacement. Also, when performing greedy matching with replacement, the order of matches does not matter, while greedy matching without replacement will produce different results depending on the order that cases are matched. Matching with replacement performs better when the number of available matches is small (Rosenbaum, 1989), but the difference between these methods tends to disappear as the size of the pool of available matches increases.

***One-to-one, fixed ratio, or variable ratio matching.*** When each treated case has one untreated case matched to it, the procedure is described as one-to-one matching or pair matching. One disadvantage of one-to-one matching is that it discards untreated cases even if they are appropriate matches for the treated cases, reducing the sample size to twice the number of treated cases or fewer if there is lack of common support for some treated cases. However, one-to-one matching does not result in a substantial drop of power, because the power is driven by the size of the smallest group, and the increased homogeneity of the sample increases power (Cohen, 1988).

The use of fixed ratio matching or one-to-*k* matching, where *k* specifies a fixed matching ratio larger than 1 (e.g., one-to-two, one-to-three), is not recommended in most situations, because matching to the specified number of untreated cases will occur regardless of whether enough adequate matches are available, leading to an increase in bias. Although the use of a caliper may limit the increase in bias, using a caliper with a fixed ratio larger than one-to-one may result in substantial loss of treated cases, because there may not be enough untreated cases within the caliper to satisfy the specified ratio. If matching with a ratio larger than one-to-one is desired to retain a larger sample size, it is more advantageous to use either variable ratio matching or full matching.

If each single treatment case is matched to one to several untreated cases (i.e., the number of matches varies across treated cases), the method is known as variable ratio matching or one-to-many matching. Research has shown that variable ratio matching removes more bias than one-to-one matching (Cepeda, Boston, Farrar, & Strom, 2003; Gu & Rosenbaum, 1993; Ming & Rosenbaum, 2000). Variable ratio matching is

particularly useful if the number of untreated group cases is much larger than the number of treatment cases (Ming & Rosenbaum, 2001). Variable ratio matching is known to outperform one-to-one matching for estimating the treatment effect in general conditions, but the difference in performance between these methods decreases as the number of available matches increases (Cepeda et al., 2003; Gu & Rosenbaum, 1993).

***Nearest neighbor or within-caliper matching.*** Greedy propensity score matching has been performed using either a nearest neighbor or nearest neighbor within-caliper matching strategy. Matching to the nearest neighbor consists of simply finding the untreated observation with the closest propensity score to the propensity score of each treated observation. The use of a caliper, which is a maximum distance within which matches are allowed, has been shown to improve greedy nearest neighbor matching performance. A caliper for matching is usually defined as a fraction of the standard deviation of the logit of the propensity score. Targeting to remove at least 90% of bias, Rosembaum and Rubin (1985, p. 37) used a caliper of .25 standard deviations. Using a caliper not only improves the quality of matching but also enforces common support, because treated cases without any untreated cases within its caliper are discarded. When nearest neighbor matching within a caliper is used, the closest untreated case to a treated case is only considered an acceptable match if its propensity score lies within the caliper of the treated case.

***Implementation of greedy matching.*** Two different R packages will be used to demonstrate greedy matching, because their different features may be helpful to researchers. The *MatchIt* package (Ho et al., 2011) is focused on estimating the ATT, and its *matchit* function will be used to demonstrate greedy one-to-one matching with replacement within a 0.25 caliper.

```
library(MatchIt)

greedyMatching <- matchit(psFormula, distance=data$logitPScores, m.order="largest",
    data = data, method = "nearest", ratio=1,replace=T, caliper=0.25)
```

In the code above, the argument *distance=data$logitPScores* specifies the variable that contains the logit of the propensity scores. However, providing the logit of the propensity scores is optional, because the *matchit* function is able to fit a variety of models to estimate the propensity score before performing matching. For example, if the argument *distance = "linear.logit"* is provided, the *matchit* function estimates propensity scores using logistic regression and converts the predicted probabilities into logits as shown in Equation (5.1) and then performs matching. However, the advantage of estimating the propensity score in advance of using the *matchit* function as it is done in this example is that better control of the process is obtained, including using approaches to estimate propensity scores that have not been implemented in the *matchit* function.

In the *matchit* function, the argument *method = "nearest"* in combination with *caliper = 0.*25 specifies that the greedy method is to be performed by searching for the nearest untreated observation within a caliper of 0.25 standard deviations of each treated observation. The argument *m.order = "largest"* specifies that matching should start from the treated case with the largest propensity score, rather than the smallest or random. This argument reflects the fact that greedy nearest neighbor matching does not optimize global measures of balance, and therefore different results are obtained depending on the order of matching. The *ratio=1* argument specifies one-to-one matching but can be used to specify other fixed matching ratios. The *replace = T* argument specifies matching with replacement, which allows untreated cases to be matched to more than one treated case.

The *Match* function of the *Matching* package of R (Sekhon, 2011) can perform greedy matching with fixed and variable ratio to estimate either the ATT or ATE. For estimating the ATT, matching is used to identify which untreated observations have similar values of propensity scores as treated observations. For estimation of the ATE, matches for treated observations as well as matches for untreated observations are selected. Here it will be used to implement variable ratio greedy matching with replacement to estimate the ATT.

```
library(Matching)

greedyMatching2 <- with(data, Match(Y=C0338600, Tr=childCare, X=logitPScores,

                estimand = "ATT", M = 1, caliper = 0.25, replace=TRUE, ties=TRUE))
```

In the code above, the arguments *Y, Tr,* and *X* specify the outcome, the treatment, and the propensity scores, respectively. The argument *M = 1* specifies one-to-one matching, but variable ratio matching is performed implicitly by allowing ties with the argument *ties=TRUE,* and therefore if more than one case are adequate matches to another case, all matches are included. The argument *replace=TRUE* specifies that untreated cases can be used for multiple matches, and *caliper = 0.25* sets the maximum allowed distance between a treated and an untreated case to be equal to 0.25 standard deviations.

### 5.4.2. Genetic Matching

Genetic matching (Diamond & Sekhon, 2013) minimizes a multivariate weighted distance on covariates between treated and untreated cases, where a genetic algorithm is used to choose weights that optimize postmatching covariate balance. Genetic matching can be used without including propensity scores, but propensity scores can be used by themselves or added to the list of covariates. The distance minimized by the genetic matching algorithm is the generalized Mahalanobis distance (GMD) (Diamond & Sekhon, 2013):

$$GMD(X_i, X_j, W) = (X_i - X_j)^T \left(S^{-1/2}\right)^T WS^{-1/2}(X_i - X_j) \quad (5.2)$$

In the GMD, *X* are vectors of covariates for treated case *i* and untreated case *j*, and the weight matrix *W* is included to reflect the relative importance of each covariate to optimize overall covariate balance. *W* is a diagonal weight matrix with rows and columns equal to the number of covariates. $S^{-1/2}$ is the Cholesky decomposition of *S*, the variance covariance matrix of the covariates (Sekhon, 2011). *T* indicates the transpose. The GMD can be understood as a weighted average effect size between treated and untreated groups across all covariates.

The genetic matching algorithm is available in the *Matching* package of R (Sekhon, 2011) with the *GenMatch* function. The *GenMatch* function uses the genetic algorithm to obtain weights that optimize covariate balance. The following code demonstrates genetic matching with replacement to estimate the ATT based on the linear propensity score and 31 covariates, which are stored in the *covariateData* object.

```
geneticWeights <- GenMatch(Tr=data$childCare, X=covariateData,
                           pop.size=1000, fit.func="pvals",
                           estimand="ATT", replace=T, ties=T)
```

For each generation (i.e., iteration), the genetic algorithm sets the weights in *W* to initial values (the default initial value is 1 in *GenMatch*) and generates as many *W* as the specified population size in the *pop.size* argument. Because genetic matching optimizes covariate balance asymptotically, it is important to specify a large population size for the genetic optimization. The default of the *GenMatch* function is *pop.size = 100*, which is increased to 1,000 in the following code, but larger values may be necessary. Then, the algorithm matches for each *W* in a given generation. Next, it computes the loss for each matched sample and selects the *W* corresponding to the minimum loss. The algorithm requires the specification of a loss function, which is a summary of a measure of covariate balance. The default loss function is specified in *fit.func = "pvals"*, which consists of the maximum of *p* values from Kolmogorov-Smirnov tests and paired *t* tests for all covariates. While using *p* values for covariate balance assessment is problematic because it depends on sample size, it is a good choice for defining the fit function because the sample size is fixed within the optimization (Diamond & Sekhon, 2013). If the convergence criterion is reached, the genetic algorithm returns the matched sample and corresponding *W* matrix; otherwise, it proceeds to the next generation. Details about the genetic matching algorithm are provided by Sekhon (2011). Once the matrix of weights is obtained with the *GenMatch* function, the actual matching procedure is implemented by the *Match* function using the weight matrix, shown as follows:

```
geneticMatching <- Match(Y=data$C0338600, Tr=data$childCare, X=covariateData,
                         Weight.matrix = geneticWeights, estimand = "ATT",
                         M = 1, replace=TRUE, ties=TRUE)
```

Genetic matching can also be obtained by using the *matchit* function of the *MatchIt* package, which can run *GenMatch* in the background. The following code implements one-to-many genetic matching without replacement based solely on the linear propensity score with the *matchit* function:

```
geneticMatching2 <- matchit(psFormula, distance=data$logitPScores, data = data,
                    method = "genetic", pop.size=1000, fit.func="pvals",
                    estimand="ATT", replace=T, ties=T)
```

A major strength of genetic matching is that it searchers for matches that optimize covariate balance. Sekhon and Grieve (2009, 2012) found through simulation studies that genetic matching based on covariates without using the propensity score is able to provide adequate covariate balance, and in their studies, it produced better balance than propensity score matching. In another simulation study, Diamond and Sekhon (2013) found that genetic matching on covariates provided greater bias reduction and lower root mean squared error than greedy matching using propensity scores estimated with logistic regression, random forests, and boosted regression trees, in conditions where the treatment assignment model had nonlinear and interaction terms. Therefore, genetic matching without the propensity score could be particularly useful for situations when propensity score matching fails to achieve covariate balance, or propensity score estimation results in complete separation or quasi-complete separation (Allison, 2004) of treated and untreated groups.

### 5.4.3. Optimal Matching

Optimal matching was proposed by Rosenbaum (1989) as a solution to the problem that greedy matching does not guarantee matches with the minimum total distance between treated and matched groups. Optimal matching produces matches that attain minimal total distances by using network flow optimization methods (Carré, 1979; Ford & Fulkerson, 1962). Hansen (2007) created the *optmatch* package for R, which produces one-to-one, one-to-*k*, and full matching. However, Rosenbaum (1989) cautioned that optimal one-to-one and one-to-*k* matching only guarantees minimum total distance given the constraint of the matching ratio desired. Optimal one-to-one matching is expected to outperform one-to-one greedy matching, but the differences in match quality are small when many matches are available. However, when the treated to untreated ratio is large, one-to-one optimal matching is noticeably better than one-to-one greedy matching (Gu & Rosenbaum, 1993).

The following code uses the *matchit* function of the *MatchIt* package with the argument *method = "optimal"*, which runs the *optmatch* package in the background to perform optimal one-to-one optimal matching without replacement:

```
optimalMatching <- matchit(psFormula,distance=data$logitPScores, data = data,
                    method = "optimal", ratio=1)
```

### 5.4.4. Full Matching

Full matching (Rosenbaum, 1991) matches each treated case to at least one untreated case and vice versa, without replacement. Therefore, this procedure can be viewed as a propensity score stratification where the number of strata containing at least one treated and one untreated observation is maximized. Differently from one-to-one matching with replacement and variable ratio matching with replacement, the matched sets never overlap and observations are not discarded, which allows the estimation of treatment effects and standard errors with methods appropriate for finely stratified samples (Hansen, 2007). Full matching is particularly helpful when there are large differences in the distributions of propensity scores between treated and untreated (assuming common support is still adequate): In this case, there will be many untreated cases with low propensity scores, so in the lower part of the propensity score distribution, there will be several matches for each treated case. However, in the upper part of the propensity score distribution, there will be few untreated cases to match to each treated case (Hansen, 2007; Stuart & Green, 2008). Full matching has been found to perform better than one-to-many greedy matching in terms of distance within matched sets as well as covariate balance, especially when the number of covariates is large (Gu & Rosenbaum, 1993).

The following code implements optimal full matching using the *matchit* function of the *MatchIt* package with the argument *method = "full"*, running the *optimatch* package in the background:

```
fullMatching <- matchit(psFormula,distance=data$logitPScores, data = data, method = "full")
```

Table 5.1 presents a summary of the matching methods implemented in this section, highlighting their unique characteristics. From the methods implemented, genetic, optimal, and full matching optimize distances for the entire sample, while greedy matching does not. From these methods, only genetic matching can optimize covariate balance directly, while optimal and full matching only match based on propensity scores. It is not possible to recommend a single matching algorithm implementation as superior to the others for all situations, because matching algorithm performance depends on the treated and untreated sample sizes, the degree of common support, and the distributions of propensity scores for treated and untreated. Therefore, it is best to implement multiple methods and compare covariate balance between them, as done in the next section.

## 5.5. Evaluation of Covariate Balance

There are 31 covariates in the propensity score model, but covariance balance is evaluated for the propensity score, continuous covariates, and levels of categorical covariates, so the total number of covariate balance measures obtained is 42. This section

**TABLE 5.1 • List of Matching Methods Used for Example**

| Matching Method | Summary |
|---|---|
| One-to-one greedy with replacement and caliper | Match based on closest observation without considering total distance for sample; fast to implement; replacement allows best matches to be used; caliper enforces common support |
| Variable ratio greedy with replacement and caliper | Match based on closest observation without considering total distance for sample; allows multiple matches per observation |
| Variable ratio genetic with replacement (**propensity score** [PS] + covariates) | Match optimizing loss function, which is a summary of a measure of covariate balance |
| Variable ratio genetic with replacement (PS only) | Match optimizing loss function based on balance of propensity scores, faster than matching on covariates |
| One-to-one optimal without replacement | Match minimizing global propensity score distance |
| Full matching | Match entire sample by creating strata with at least one treated and one untreated, minimizing global propensity score distance |

provides code to evaluate covariate balance with both the *MatchIt* and *Matching* packages. The following code is for covariate balance evaluation with the *MatchIt* package after one-to-one greedy matching was implemented, but similar code can be used for any matching method implemented with the *MatchIt* package, such as genetic, optimal, and full matching.

```
balance.greedyMatching <- summary(greedyMatching, standardize=T)
```

The next piece of code is to evaluate covariate balance with the *MatchBalance* function of the *Matching* package after genetic matching is implemented, but it can be used for greedy matching as well. The argument *match.out = geneticMatching* specifies the object generated by the *Match* function containing the matches, so to use this code to evaluate balance for the greedy matching shown earlier, the only change needed is *match.out = greedyMatching2*.

```
balance.geneticMatching <- MatchBalance(psFormula, data = data,
                                        match.out = geneticMatching, ks = F, paired=F)
```

The comparison of covariate balance achieved by different matching methods is shown in Table 5.1. None of the matching methods produced absolute standardized

mean differences lower than 0.1 for all covariates, but three produced differences lower than 0.25 standard deviations for all covariates. It is interesting to note that genetic matching with the propensity score performed better than genetic matching with the propensity score plus covariates, but this may not always be the case, so comparing both methods is recommended. At this point, the researcher may proceed with the analysis or go back and try to improve covariate balance. This decision depends on the researcher's chosen target for adequate balance, and different recommendations for what is acceptable covariate balance are presented in Chapter 1. If the target is to obtain standardized mean differences below 0.1 for all covariates, the researcher could attempt to change the propensity score model or change the propensity score estimation method and perform matching again. However, if the target is to obtain standardized mean differences below 0.25, then three of the matching methods shown in Table 5.2 performed adequately. In the next section, the matched data sets from variable ratio genetic with replacement (propensity score + covariates), variable ratio genetic matching with replacement (propensity score only), and full matching will be used to demonstrate a variety of treatment effect estimation methods.

**TABLE 5.2 ● Comparison of Covariate Balance Across Matching Methods**

| Matching Method | Maximum Absolute Standardized Mean Difference | Covariates With Absolute Standardized Mean Difference Above 0.1, *n* (%) |
|---|---|---|
| One-to-one greedy with replacement and caliper | 0.21 | 11 (26.1) |
| Variable ratio greedy with replacement and caliper | 0.30 | 4 (9.5) |
| Variable ratio genetic with replacement (PS + covariates) | 0.23 | 12 (28.6) |
| Variable ratio genetic with replacement (PS only) | 0.13 | 8 (19.0) |
| One-to-one optimal without replacement | 0.28 | 11 (26.1) |
| Full matching | 0.26 | 4 (9.5) |

Note: PS = propensity score. Results based on 42 standardized mean differences, which include the propensity score, continuous covariates, and values of categorical covariates.

## 5.6. Estimation of Treatment Effects

Treatment effect estimation methods with propensity score matched samples may differ depending on matching method, nature of the outcome, and whether the researcher decides to use parametric or nonparametric methods. The implementation of propensity score methods separates the design part of the analysis from the analysis of outcome (Rubin, 2005, 2007, 2008). In this chapter, the design part of the analysis consists of propensity score matching. Because of this separation, any parametric or nonparametric method can be used for the analysis of outcomes. In many academic fields, researchers have strong traditions of using specific parametric models for certain outcomes. Ho et al. (2007) recommend using the same parametric models with the propensity score matched samples, because they account for theoretical relationships well known in the field and provide additional bias reduction. An example of a complex parametric model (i.e., structural equation modeling) used with propensity score matching is provided in Chapter 8. In the current chapter, the focus is on simple matching estimators proposed by Abadie and Imbens (2002, 2006), the bias-corrected Abadie and Imbens estimator, and treatment effect estimation based on mean differences and linear regression (Schafer & Kang, 2008).

There is disagreement on whether propensity score matching produces clustering effects that should be accounted for in the outcome analysis. Schafer and Kang (2008) argued that matched samples should be treated as independent data because matching does not produce correlations between outcomes of matched individuals. Stuart (2010) supported Schafer and Kang's argument by adding that propensity score matching does not guarantee that covariate values are the same for matched pairs, only that covariate distributions are similar for treated and matched groups. In contrast, Austin (2011a) argued that because covariates that have similar distributions for matched and treated groups are related to the outcomes, the distributions of outcomes will be more similar for treated and matched samples than from randomly selected samples. With a simulation study of the effect of binary treatments on binary outcomes, Austin found that analyses treating the matched pairs as dependent rather than independent resulted in Type I error rates closer to the .05 alpha level, coverage of 95% confidence intervals closer to 95%, and narrower confidence intervals. Therefore, he recommended that matched samples be analyzed with methods for dependent samples. Although the differences between the results with independent and dependent sample methods in Austin's simulation study were small, they were consistently in favor of treatment matched pairs as dependent samples. Given that additional research is needed on this issue, this chapter includes methods for estimating standard errors for treatment effects that represent both sides of this debate.

The first estimator of the treatment effect appropriate for matching discussed here is the Abadie-Imbens simple matching estimator (Abadie & Imbens, 2002, 2006).

Under Rubin's potential outcomes framework, the treatment effect for a single case $i$ is $\tau_i = Y_i^1 - Y_i^0$, where $Y_i^1$ is the potential outcome under the treatment condition and $Y_i^0$ is the potential outcome under the untreated condition (see Chapter 1 for details). Similarly to a missing data problem, propensity score matching can be seen as a method to provide imputations for the potential outcomes:

$$\hat{Y}_i^1 = \begin{cases} Y_i \text{ if } Z_i = 1 \\ \dfrac{1}{M_i} \sum_{j \in J_M(i)} Y_j \text{ if } Z_i = 0 \end{cases} \tag{5.3}$$

$$\hat{Y}_i^0 = \begin{cases} Y_i \text{ if } Z_i = 0 \\ \dfrac{1}{M_i} \sum_{j \in J_M(i)} Y_j \text{ if } Z_i = 1 \end{cases} \tag{5.4}$$

In Equations (5.3) and (5.4), $Y_i$ is the observed outcome of case *i,* which was either exposed to the treated ($Z_i = 1$) or the untreated ($Z_i = 0$) condition. $M_i$ is the total number of matches for each case; the total set of matches for case $i$ is $J_M(i)$, and the outcome of each matched case $j$ is $Y_j$. Given the imputed potential outcomes obtained through matching, the ATE can be estimated as the average of the differences between the imputed potential outcomes under the treated and untreated conditions for all $n$ cases, while the ATT can be estimated taking the average difference only for the $n_1$ treated cases, as shown below (Abadie & Imbens, 2002, 2006):

$$ATE = \frac{1}{n} \sum_{1}^{n} \left( \hat{Y}_i^1 - \hat{Y}_i^0 \right) \tag{5.5}$$

$$ATT = \frac{1}{n_1} \sum_{i \in Z_1}^{n1} \left( \hat{Y}_i^1 - \hat{Y}_i^0 \right) \tag{5.6}$$

Estimates obtained with the matching estimators in Equations (5.5) and (5.6), as well as standard errors estimated with the Abadie and Imbens (2006) method, are provided by the *Matching* package (Sekhon, 2011). The Abadie-Imbens estimators will be demonstrated with the matched sample obtained with genetic matching with the propensity score and 31 covariates. The Abadie-Imbens matching estimators were proposed for general multivariate matching but can be used for propensity score matching. The following code for estimating treatment effects with the Abadie-Imbens estimators using the sample from genetic matching is also applicable to data sets obtained with greedy matching. The *geneticMatching* object was created earlier by sequentially using the functions *GenMatch* and *Match* of the *Matching* package.

*summary(geneticMatching)*

This code produces the following output:

```
Estimate... 3.7664
AI SE...... 2.6266
T-stat..... 1.4339
p.val...... 0.1516

Original number of observations.............. 1209
Original number of treated obs............... 107
Matched number of observations............... 107
Matched number of observations (unweighted).  107
```

The estimate of the ATT is 3.766 (*SE* = 2.626, *p* = .152), indicating that mothers who had a job that provided or subsidized child care did not breastfeed their child longer than if they had a job that did not provide this benefit. The output above indicates that the number of mothers in the treatment condition is 107, and they are matched to 107 untreated mothers. Therefore, for this example, although the algorithm is set to allow multiple matches per treated case, it produced one-to-one matching, because only one untreated observation was identified for each treated observation that optimized balance in the propensity score as well as 31 covariates.

Abadie and Imbens (2002) showed that the matching estimator will be biased if the matching is not exact, but this bias can reduced by regressing the outcomes on covariates only with the matched data. The *Matching* package allows bias adjustment, which is accomplished with the *Match* function by adding the *BiasAdjust=T* argument and *Z=covariateData,* which specifies the data set containing the covariates that will be used for bias adjustment.

```
geneticMatchingBA <- Match(Y=data$C0338600, Tr=data$childCare, X=covariateData,
                    BiasAdjust=T, Z=covariateData, Weight.matrix = geneticWeights,
                    estimand = "ATT", M = 1, replace=TRUE, ties=TRUE)
```

The code for multivariate genetic matching with bias adjustment for all covariates shown above produces estimates that can be extracted with the *summary* function:

*summary(geneticMatchingBA)*

```
Estimate... 4.352
AI SE...... 2.7694
T-stat..... 1.5714
p.val...... 0.11608
```

The estimate of the ATT is 4.352 (*SE* = 2.769, $p$ = .116), showing that bias adjustment did not alter the conclusion obtained without bias adjustment that there is no treatment effect.

The *MatchIt* package (Ho et al., 2011) does not estimate treatment effects directly, but it provides a matched data set with case weights that can be used to estimate the ATT. In one-to-one matching without replacement, all case weights are 1. However, if fixed ratio (greater than one-to-one) or variable ratio matching is done, weights for treated cases are 1 but weights for all untreated cases matched to a treated case are the inverse of the total number of matches the treated case received Also, if matching is done with replacement, case weights for each untreated case are summed across the multiple matched groups in which it was included. Finally, the weights of the matched cases are multiplied by the ratio of the total number of matched cases and total number of treated cases, which scales the untreated weights to sum to the number of matched cases (Ho, Imai, King, & Stuart, 2014). The following equation implements the calculation of weights described above:

$$w_i = \begin{cases} 1 \text{ if } Z_i = 1 \\ \dfrac{n_0}{n_1} \displaystyle\sum_{m=1}^{n_i} \dfrac{1}{M_m} \text{ if } Z_i = 0 \end{cases} \tag{5.7}$$

where $n_i$ is the number of treated cases that case $i$ was matched to, $M_m$ is the total number of matches including case $i$ that each treated case received, $n_0$ is the total number of matched cases, and $n_1$ is the total number of treated cases. If a caliper is used, treated cases without untreated cases within their calipers are dropped before weights are calculated.

Using the matched sample obtained with variable ratio genetic matching with replacement based only on the propensity score, the following code demonstrates the estimation of treatment effects as a difference between weighted means. The function *match.data* of the *MatchIt* package was used to extract the *data.geneticMatching2* matched data set, which is then analyzed with the *survey* (Lumley, 2004) package. First, the *svydesign* function is used to specify the name of the data to be analyzed and the *weights* variable that contains the weights resulting from the variable ratio with replacement matching method.

```
data.geneticMatching2= match.data(geneticMatching2)

library(survey)

design.geneticMatching2 <- svydesign(ids=~1, weights=~weights,
                                     data=data.geneticMatching2)
```

Then, the *svyglm* is used with the formula *C0338600~childCare* to fit the simple regression model $Y_i = \beta_0 + \beta_1 Z_i + \varepsilon_i$, where $\beta_1$ is the treatment effect.

```
model.geneticMatching2 <- svyglm(C0338600~childCare, design.geneticMatching2,
                                 family=gaussian())
```

The code above applies weights as defined in Equation (5.7) to the outcomes but does not implement any method to account for any clustering effects due to matching. The treatment effect is shown with the *summary* function:

```
summary(model.geneticMatching2)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)     8.916      1.581   5.641  5.5e-08 ***
childCareTRUE   4.607      2.679   1.720    0.087 .
```

With variable ratio genetic matching with replacement, the ATT estimate is 4.607 ($SE$ = 2.679, $p$ = .087), which is similar to the previous two estimates.

Next, the ATT will be estimated using the data set extracted with the *matched.data* function from the *fullMatching* object, which contains the results of full matching performed earlier with the *matchit* function. Full matching does not drop any observations, so the sample size for the estimation of treatment effects is much larger than with one-to-one matching.

```
data.fullMatching= match.data(fullMatching)
```

After the matched data set is extracted, the *svydesign* function of the *survey* package is used to specify the data and weights to be used to fit the outcome model.

```
design.fullMatching <- svydesign(ids=~1, weights=~weights,
                                 data=data.fullMatching)
```

The following code uses a regression model fit with the *svyglm* function of the *survey* package to estimate the ATT as the difference between weighted means, with weights obtained according to Equation (5.7).

```
model.fullMatching <- svyglm(C0338600~childCare, design.fullMatching, family=gaussian())
summary(model.fullMatching)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   10.0428     0.8887  11.300   <2e-16 ***
childCareTRUE  3.4806     2.3346   1.491    0.136
```

The analysis results with the data set obtained with full matching show a treatment effect estimate (ATT = 3.481, *SE* = 2.335, *p* = .136) that is lower than those estimated with the methods presented previously, but it agrees with the other results with respect to the estimate not being statistically significant.

The use of a without-replacement matching strategy provides nonoverlapping matched pairs, which allows the use of a design-based method to account for clustering effects. Abadie and Imbens (2008) showed that bootstrapping is not an appropriate estimator for the standard error of treatment effects when matching with replacement is performed. However, Austin and Small (2014) found that bootstrapping matched pairs is an effective method to estimate standard errors when matching without replacement is used. In the following code, standard errors are obtained by bootstrapping the clusters of observations formed by the full matching algorithm. The argument *ids=~subclass* to the *svydesign* function specifies the cluster ids. This strategy is consistent with Austin's recommendation to adjust for pair effects when estimating standard errors from matched data.

```
design.fullMatching2 <- svydesign(ids=~subclass, weights=~weights,
                                  data=data.fullMatching)
```

The *as.svrepdesign* function specifies that bootstrapping will be performed with 1,000 replications:

```
design.fullMatching2 = as.svrepdesign(design.fullMatching2, type="bootstrap",
                                      replicates=1000)
model.fullMatching2 <- svyglm(C0338600~childCare, design.fullMatching2, family=gaussian())
summary(model.fullMatching2)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    10.043      0.880  11.413  <2e-16 ***
childCareTRUE   3.481      2.385   1.459   0.148
```

The only difference between the results below and the previous ones is the standard error estimation method, but the difference between the standard errors is very small. The estimate obtained (ATT = 3.481, SE = 2.385, p = .148) for the effect of the availability of company-provided or subsidized child care on length of breastfeeding is not statistically significant.

## 5.7. Sensitivity Analysis

Sensitivity analysis consists of examining what magnitude of hidden bias would change inferences about a treatment effect. Rosenbaum (2002) proposed a nonparametric sensitivity analysis method for continuous and ordinal outcomes based on the Wilcoxon signed ranks test. Rosenbaum's sensitivity analysis method is briefly described here and demonstrated with the rbounds package in R.

Rosenbaum's sensitivity analysis is based on the principle that if two cases have the same values on observed covariates but different probabilities of treatment assignment, the odds ratio of these cases receiving the treatment is

$$\frac{\pi_j/(1-\pi_j)}{\pi_k/(1-\pi_k)} = \frac{\pi_j(1-\pi_k)}{\pi_k(1-\pi_1)} \tag{5.8}$$

If there is an unobserved confounder, the odds ratio will be larger than 1 and smaller than a constant $\Gamma$ (gamma) that measures the degree of departure from the absence of hidden bias.

$$\frac{1}{\Gamma} \leq \frac{\pi_j(1-\pi_k)}{\pi_k(1-\pi_1)} \leq \Gamma \tag{5.9}$$

Therefore, the value of $\Gamma$ can be manipulated to evaluate how large it has to be for inferences about the significance of the treatment effect to change. If $\Gamma$ has to attain very high values for inferences to change, then it is possible to conclude that the treatment effect is insensitive to hidden bias. Rosenbaum's sensitivity analysis consists of computing p values of the lower and upper bounds of the Wilcoxon signed rank statistic for the outcome difference between treated and untreated groups, under null hypotheses with increasing values of $\Gamma$.

The rbounds package was designed to work together with the Matching package to implement Rosenbaum's sensitivity analysis method. It can handle matched outcomes obtained with packages other than Matching, but it can currently handle only one-to-one and fixed ratio matching. Therefore, sensitivity analysis with one-to-one genetic matching will be demonstrated but not with full matching. The following line of code uses the psens function of the rbounds package to implement the sensitivity analysis with the geneticMatching object obtained previously with the Match function of the Matching package, by varying the sensitivity parameter gamma from 1 to a maximum given by Gamma = 3, in increments of 0.1 specified by the GammaInc argument:

*psens(geneticMatching, Gamma=3, GammaInc=.1)*

```
Rosenbaum Sensitivity Test for Wilcoxon Signed Rank P-Value
 Unconfounded estimate .... 0.1305
Gamma Lower bound Upper bound
  1.0      0.1305      0.1305
  1.1      0.0686      0.2224
  1.2      0.0343      0.3311
  1.3      0.0166      0.4456
  1.4      0.0077      0.5560
  1.5      0.0035      0.6553
  1.6      0.0016      0.7396
  1.7      0.0007      0.8080
  1.8      0.0003      0.8613
  1.9      0.0001      0.9017
  2.0      0.0001      0.9315
  2.1      0.0000      0.9529
  2.2      0.0000      0.9681
  2.3      0.0000      0.9786
  2.4      0.0000      0.9858
  2.5      0.0000      0.9906
  2.6      0.0000      0.9939
  2.7      0.0000      0.9960
  2.8      0.0000      0.9974
  2.9      0.0000      0.9984
           0.0000      0.9990
```

In this particular example, the results show that although the *p* value assuming no hidden bias is not statistically significant, a value of equal to 1.2 or larger could lead to a significant *p* value, and therefore the conclusion that there is no effect of availability of company-provided or company-subsidized child care on length of breastfeeding is vulnerable to hidden bias. If the lower bound of the *p* value did not overlap the significance level even with $\Gamma$ as high as 3, then there would be evidence of the results not being sensitive to hidden bias.

**TABLE 5.3 ● Summary of Main Functions Used in This Chapter**

| Package | Function | Objective |
|---|---|---|
| **stats** | *glm* | Estimate propensity scores with logistic regression |
| **MatchIt** | *matchit* | Implement greedy matching and as interface for genetic, optimal, and full matching |
| **MatchIt** | *match.data* | Extract the matched data set from the object created with the *matchit* function |
| **Matching** | *GenMatch* | Obtain covariate weights for genetic matching |

| Package | Function | Objective |
|---|---|---|
| **Matching** | *Match* | Implement genetic matching with weights provided by the *GenMatch* function, as well as greedy matching |
| **Matching** | *MatchBalance* | Evaluate covariate balance of matching results provided by the *Match* function |
| **survey** | *svydesign* | Create an object that specifies the data to be analyzed, strata ids, cluster ids, and weights |
| **survey** | *as.svrepdesign* | Add replication weights to an object created with *svydesign* function, to allow bootstrapping |
| **survey** | *svyglm* | Estimate treatment effect with a generalized linear model |
| **rbounds** | *psens* | Implement Rosenbaum's sensitivity analysis method |

## 5.8. Conclusion

Both multivariate and propensity score matching have been used and studied extensively since seminal work by Rubin in the late 1970s and early 1980s. Therefore, discipline-specific preferences have been developed with respect to how matching is used. For example, in the economics field, matching is performed primarily with replacement (Abadie & Imbens, 2006; Imbens, 2004; Imbens & Wooldridge, 2009), while in medicine, it is performed most commonly without replacement (Austin, 2008; Austin & Small, 2014). Although matching with replacement is able to produce better covariate balance than matching without replacement, it complicates statistical analysis, particularly with respect to estimation of standard errors.

With large samples with many untreated cases available for each treated case, matching with or without replacement makes little difference in covariate balance. This is particularly true when algorithms that optimize covariate balance, such as the genetic and optimal algorithms, are being used. However, as shown in the covariate balance evaluation comparison in Table 5.2, despite the theoretical advantages of optimal, full, and genetic matching over greedy matching, it is not always possible to predict how each matching algorithm will actually perform with respect to covariate balance for a particular sample. Therefore, it is recommended that multiple matching algorithms are implemented and the results are compared. It is common in applications of propensity score matching to report results based on multiple methods, which provides some evidence on how sensitive the conclusions are to matching algorithm choice.

Implementation of a propensity score matching method is more complicated and involves many choices of algorithms and tuning parameters than implementation of

propensity score weighting shown in Chapter 3. This begs the question of whether there are situations when using matching would be preferable over propensity score weighting. First, there are situations when matching is able to produce better covariate balance than weighting, due to a combination of factors, such as differences in sample sizes and the distributions of propensity scores of treated and untreated. Second, when there are difficulties in estimating propensity scores due to nonconvergence or quasi-separation of treated and untreated groups, matching based on covariates without the propensity score with the genetic matching algorithm is possible. Finally, there are situations when a stakeholder of an evaluation project is willing to accept matching as a quasi-experimental design method but not weighting. For example, the current standards of the What Works Clearinghouse (U.S. Department of Education, Institute of Education Sciences, & What Works Clearinghouse, 2013), which is a government organization that reviews educational research, describes matching as a method of quasi-experimental analysis but not weighting. Therefore, if the evaluators of an educational intervention are aiming to obtain a favorable review from the What Works Clearinghouse but cannot perform an experimental design, matching would be an acceptable option.

## Study Questions

1. What is the difference between the common support requirements for matching for estimating the ATT and the ATE?
2. Why is it advantageous to match based on the logit of the propensity score rather than the propensity score itself?
3. How is greedy matching performed?
4. How is optimal matching performed?
5. How is genetic matching performed?
6. What is the role of the generalized Mahalanobis distance in genetic matching?
7. In what situation would greedy matching be expected to perform as well as optimal matching?
8. What is the advantage of optimal matching over greedy matching?
9. What is the advantage of genetic matching over optimal and greedy matching?
10. What is the expected difference in performance between one-to-one, one-to-many, and variable ratio matching?
11. How is a caliper used in matching, and what is the advantage of using it?

12. How can common support be strictly enforced in propensity score matching?
13. What is the expected difference in performance between matching with and without replacement?
14. How is full matching performed?
15. Which matching methods require the use of weights and why?
16. Why do some researchers argue that matched samples should be treated as independent samples?
17. Why do some researchers argue that matched samples should be treated as related samples?
18. What is the Abadie-Imbens simple estimator for matched samples?
19. How can treatment effects for matched samples be estimated with regression?
20. What is the objective of sensitivity analysis?
21. What is the role of the odds ratio and the gamma constant in Rosenbaum's sensitivity analysis?