Inhaltsverzeichnis

1 Kovariatenmatching

Im folgenden werden wir die Schätzung kausaler Effekte nach Kovariatenmatching betrachten. Wir betrachten zwei Methoden des Kovariatenmatchings: exaktes Matching und Nearest Neighbor Matching.

1.1 Lernziele und Vorgehen

Sie können Kovariatenmatching mithilfe des Pakets `MatchIt` durchführen, das Ergebnis des Matchings beurteilen, auf Basis der gematchten Stichproben Schätzer für kausale Effekte berechnen und diese interpretieren.

Wir werden dafür wie folgt vorgehen:

Lernziele und Vorgehen

1. Deskriptive Datenanalyse
2. Kovariatenmatching mit `MatchIt`
3. Exaktes Matching
   a) Durchführung
   b) Imputierte Potential Outcomes
   c) Effektschätzung
4. Nearest Neighbor Matching
   a) Durchführung
   b) Effektschätzung

1.2 Vorbereitung

Wir benötigen in dieser Sitzung die folgenden Pakete.

```
if (!require("MatchIt")) install.packages("MatchIt")
if (!require("psych")) install.packages("psych")
if (!require("ggplot2")) install.packages("ggplot2")
```

Wir arbeiten mit dem Datensatz `bdi_data.rda`. Arbeitsverzeichnis setzen und Datensatz öffnen:

```
setwd("C:/Users/me/myworkingdirectory")
load("bdi_data.rda")
```

Der Datensatz `bdi_data.rda` enthält die Daten von DepressionspatientInnen einer Hochschulambulanz (fiktiver Datensatz!). Zusätzlich zu ihrer Therapie erhalten einige PatientInnen eine achtsamkeitsbasierte Intervention (`group=1`). PatientInnen in der Kontrollgruppe (`group=0`) erhielten nur die Standardtherapie. Vor und nach der Intervention wird der BDI (*Beck Depression Inventory*) erhoben (`bdi1`, `bdi2`). Als Outcome soll die Depression nach der Intervention (`bdi2`) untersucht werden. Die Depressionswerte vor der Intervention (`bdi1`) sollen als Kovariante dienen. Als weitere Kovariaten wurden die folgenden Variablen erhoben:

- Alter (`age`)
- kognitive Leistungsfähigkeit (`cogn`)
- Anzahl bisheriger Therapiesitzungen (`sess`)

- Vorliegen einer Substanzabhängigkeit (`addic`)

```
str(bdi_data)
```

```
'data.frame':   255 obs. of  7 variables:
 $ bdi1 : num  43 33 35 27 32 21 19 25 17 35 ...
 $ bdi2 : num  27 33 25 31 33 19 14 18 13 35 ...
 $ age  : num  28 41 25 46 28 35 35 29 32 42 ...
 $ sess : num  24 16 13 24 19 22 16 36 23 20 ...
 $ cogn : num  94 73 85 74 81 90 98 86 98 72 ...
 $ addic: num  0 0 0 0 0 0 0 0 0 0 ...
 $ group: num  1 1 1 1 1 1 1 1 1 1 ...
```

1.3 Deskriptiver Vergleich der Gruppen

Mithilfe des Befehls `describeBy()` aus dem Paket `psych()` lassen wir uns gruppenspezifische Kennwerte für unsere Kovariaten ausgeben:

```
library(psych)
describeBy(bdi_data, group = bdi_data$group)
```

```
 Descriptive statistics by group
group: 0
      vars   n  mean    sd median trimmed   mad min max range skew kurtosis
bdi1     1 179 20.96 12.14     20   20.62 11.86   0  55    55 0.24    -0.39
bdi2     2 179 23.74 12.06     24   23.70 11.86   0  58    58 0.08    -0.47
age      3 179 33.83  7.54     34   33.81  7.41  18  53    35 0.05    -0.43
sess     4 179 15.36 12.28     14   14.35 14.83   0  53    53 0.56    -0.35
cogn     5 179 91.96 15.53     90   91.99 14.83  50 140    90 0.00     0.15
addic    6 179  0.42  0.50      0    0.41  0.00   0   1     1 0.30    -1.92
group    7 179  0.00  0.00      0    0.00  0.00   0   0     0  NaN      NaN
       se
bdi1  0.91
bdi2  0.90
age   0.56
sess  0.92
cogn  1.16
addic 0.04
```

```
group 0.00
------------------------------------------------------------
group: 1
      vars  n  mean    sd median trimmed   mad min max range  skew kurtosis
bdi1     1 76 30.96  9.64   32.0   31.10  7.41   4  53    49 -0.22     0.17
bdi2     2 76 23.04  7.98   23.5   23.13  9.64   1  38    37 -0.15    -0.46
age      3 76 35.82  6.96   35.5   35.69  8.15  23  51    28  0.14    -0.92
sess     4 76 20.28 10.32   20.0   20.21 10.38   0  45    45  0.12    -0.48
cogn     5 76 82.74 10.36   82.0   82.68  9.64  60 109    49  0.21    -0.25
addic    6 76  0.42  0.50    0.0    0.40  0.00   0   1     1  0.31    -1.93
group    7 76  1.00  0.00    1.0    1.00  0.00   1   1     0   NaN      NaN
       se
bdi1  1.11
bdi2  0.92
age   0.80
sess  1.18
cogn  1.19
addic 0.06
group 0.00
```

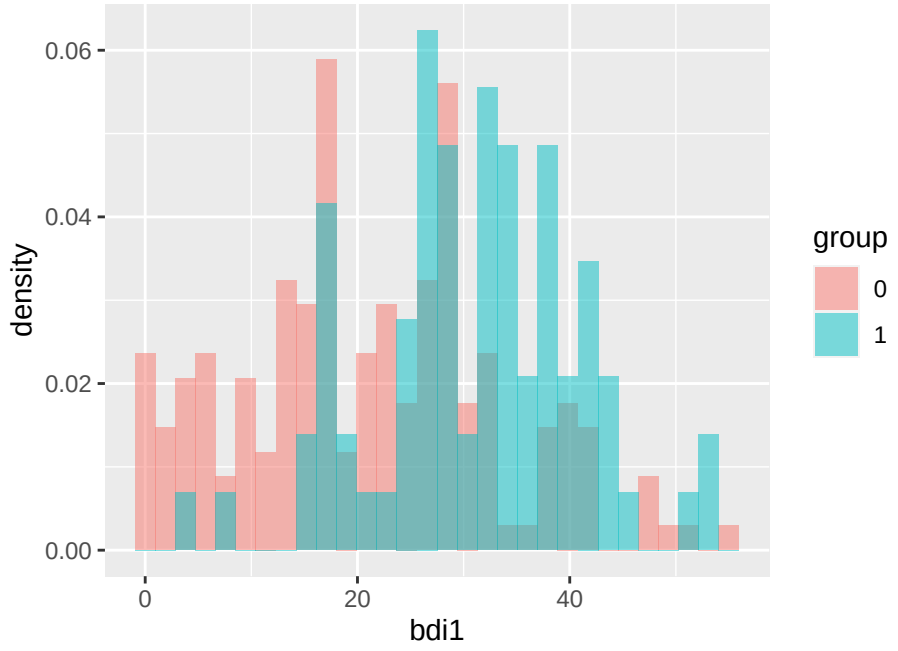
Wir stellen fest, dass sich die beiden Gruppen hinsichtlich des Durchschnitts der Outcome-Variable `bdi2` kaum unterscheiden. Sie unterscheiden sich jedoch deutlich in allen Kovariaten, mit Ausnahme der Variable `addic`. Insbesondere ist auffällig, dass die durchschnittliche Depression **vor** der Intervention in der Kontrollgruppe deutlich niedriger ist als in der Behandlungsgruppe. Diese Variable sollte daher bei einer Schätzung des kausalen Effekts unbedingt als Kovariate berücksichtigt werden.

1.3.1 Exkurs: Grafische Veranschaulichung

Zusätzlich können wir die Verteilungen der Kovariaten in den beiden Gruppen graphisch darstellen, z.B. mit dem `ggplot()`-Befehl aus dem Paket `ggplot2`. Hierfür muss die Gruppenvariable `group` als Faktor gespeichert sein. Da wir die Variable später als numerische Variable im Datensatz benötigen, speichern wir unseren Datensatz zunächst unter einem anderen Namen:

```r
# Lade Paket `ggplot2`
library(ggplot2)
# Abspeichern des Datensatzes unter anderem Namen
plot_data <- bdi_data
```

```r
# Konvertierung der Variable `group` in einen Faktor
plot_data$group <- as.factor(plot_data$group)
# Plotten der Verteilung einer Kovariate nach Gruppen (hier: Kovariate `bdi1` (pretest))
ggplot(plot_data, aes(bdi1, fill = group)) +
  geom_histogram(alpha = 0.5, aes(y = ..density..), position = "identity")
```



Wir sehen, dass die Verteilung der Depressionswerte vor der Intervention in der Kontrollgruppe (bis auf einige Ausreißer) weiter links (also im Bereich geringerer Depression) liegt als in der Interventionsgruppe. Die beiden Gruppen unterscheiden sich also deutlich in den Pretest-Werten, sodass es ratsam ist, `bdi1` als Kovariate zu berücksichtigen. Analoge Darstellungen ließen sich für die anderen potentiellen Kovariaten erstellen.

1.4 Gruppenunterschied im Outcome

Der Prima-Facie Effekt (PFE) kann anhand des einfachen (nicht adjustierten) Gruppenunterschieds im Outcome `bdi2` berechnet werden.

```r
lm(bdi2 ~ group, data = bdi_data)
```

```
Call:
lm(formula = bdi2 ~ group, data = bdi_data)

Coefficients:
(Intercept)        group
     23.737       -0.698
```

Der PFE beträgt −0.698. Dieser Effekt ist nicht für Kovariaten adjustiert und entspricht damit mit hoher Wahrscheinlichkeit nicht dem wahren Treatment-Effekt!

2 Kovariatenmatching mit `MatchIt`

- Im Paket `MatchIt` ist die Hauptfunktion `matchit()` enthalten
  - Matching auf Gruppenvariable `X` anhand von Kovariaten (`Z1`, `Z2`, …)
  - Angabe in Formelschreibweise (`X ~ Z1 + Z2`)
  - Über Argument `method` Algorithmen auswählbar, z. B. `'exact'`, `'nearest'`
  - Distanzmetriken wählbar über Argument `distance`
  - Es gibt auch ein Argument für die Wahl des Calipers (`caliper`). Wir werden dieses aber im Folgenden nicht ansprechen, da wir erstmal die Default-Einstellung nutzen werden.
- Voraussetzungen
  - **Gruppenvariable** soll **numerisch** sein (kein Faktor)
  - Kodierschema `IG=1`, `KG=0` sollte genutzt werden
- Vorgehen
  - Ausgabe in Objekt ablegen (wie Modell in Regression)
  - Mit verschiedenen Funktionen (z. B. `summary()`) darauf zugreifen)

2.1 Befehl mit Argumenten

Funktion	Bedeutung
`matchit(`	Sucht Matchingpartner für
`treat ~ x1 + x2,`	die Treatmentvariable (`treat`) mit einer oder mehrerer Kovariaten (`x1 + x2 + ...`)
`data,`	aus dem Datensatz `data`,

Funktion	Bedeutung
`method,`	mit nearest neighbor matching (`"nearest"`) als Default und bspw. exaktem Matching (`"exact"`) oder optimal matching (`"optimal"`) als weitere Optionen
`distance,`	mit verschiedenen Distanzmaßen wie beispielsweise `mahalanobis`,
`ratio,`	mit z.B. `ratio = 1` (d.h., `1:1`) oder anderen Ratios für die Anzahl der Personen in der Treatmentgruppe zu der Anzahl der zu matchenden Personen in der Kontrollgruppe
`replace)`	mit `FALSE` als Default (ohne Zurücklegen) oder `TRUE` (mit Zurücklegen)

Details siehe `?matchit`.

3 Exaktes Matching

Im Folgenden wenden wir das exakte Matching auf Basis der beiden Kovariaten `sess` und `addic` an. Bei dieser Methode werden die Argumente für die Distanzmetrik (`distance`) und das Verhältnis (`ratio`) nicht benötigt. Der Grund dafür ist, dass bei einem exakten Matching alle Teilnehmer miteinander abgeglichen werden, die identische Werte auf den Kovariaten aufweisen. Teilnehmer, für die kein entsprechender Matchingpartner mit identischen Werten gefunden wird, werden aus dem Matchingprozess ausgeschlossen.

```
library(MatchIt)
exakt <- matchit(group ~ sess + addic, data = bdi_data,
                 method = "exact")
exakt
```

```
A matchit object
 - method: Exact matching
 - number of obs.: 255 (original), 182 (matched)
 - target estimand: ATT
 - covariates: sess, addic
```

```
summary(exakt)
```

```
Call:
matchit(formula = group ~ sess + addic, data = bdi_data, method = "exact")

Summary of Balance for All Data:
      Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean eCDF Max
sess        20.2763       15.3631          0.4761     0.7064    0.1107   0.2568
addic        0.4211        0.4246         -0.0071          .    0.0035   0.0035

Summary of Balance for Matched Data:
      Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean eCDF Max
sess        18.9118       18.9118              -0     0.9954         0        0
addic        0.3971        0.3971              -0          .         0        0
      Std. Pair Dist.
sess                0
addic               0

Sample Sizes:
              Control Treated
All            179.        76
Matched (ESS)   52.07      68
Matched        114.        68
Unmatched       65.         8
Discarded        0.         0
```

- Exaktes Matching erzeugt Klassen (hier: 42) mit identischen Werten auf den angegebenen Kovariaten **Z**. Personen ohne identische/n PartnerInnen werden entfernt (`Unmatched`). Aus der Treatmentbedingung wurde hier also für 68 von insgesamt 76 Personen ein/e Matchingpartner/in gefunden. In der Kontrollbedingung konnten 114 von 179 Personen gematched werden.
- Die einzelnen Klassen können unterschiedlicher Größe sein und eine unterschiedliche Zahl von Personen aus jeder Gruppe beinhalten. Die Größe und Anzahl der Gruppenzugehörigkeiten der Personen in jeder Klasse können z.B. mit dem Befehl `summary(exakt)` ausgegeben werden. Wenn Sie sich die `summary` aufrufen, sehen Sie zum Beispiel, dass in der ersten Subklasse je 2 Personen in der Treatment- und der Kontrollgruppe sind. Diese 4 Personen haben also identische Werte auf den beiden Kovariaten.

- Die Größe der Klassen und Anzahl von ProbandInnen in der Kontroll- bzw. Treatmentgruppe in jeder Klasse muss bei der Berechnung der Treatment-Effekte berücksichtigt werden (s.u.).

Zur Effektschätzung wird der Datensatz gematchter Personen inklusive Informationen zur

`subclass` Zuordnung zunächst extrahiert. Dies erfolgt anhand des Befehls `match.data()`, welcher auf das angelegte `matchit`-Objekt zugreift.

```
exakt_data <- match.data(exakt)
str(exakt_data)
```

```
Classes 'matchdata' and 'data.frame':   182 obs. of  9 variables:
 $ bdi1    : num  43 33 35 27 32 21 19 17 35 28 ...
 $ bdi2    : num  27 33 25 31 33 19 14 13 35 24 ...
 $ age     : num  28 41 25 46 28 35 35 32 42 36 ...
 $ sess    : num  24 16 13 24 19 22 16 23 20 31 ...
 $ cogn    : num  94 73 85 74 81 90 98 98 72 76 ...
 $ addic   : num  0 0 0 0 0 0 0 0 0 0 ...
 $ group   : num  1 1 1 1 1 1 1 1 1 1 ...
 $ weights : num  1 1 1 1 1 1 1 1 1 1 ...
 $ subclass: Factor w/ 42 levels "1","2","3","4",..: 1 2 3 1 4 5 2 6 7 8 ...
 - attr(*, "weights")= chr "weights"
 - attr(*, "subclass")= chr "subclass"
```

Der Datensatz `exakt_data` enthält nur noch die 182 erfolgreich gematchten Personen. Die letzte Variable `subclass` gibt an, welcher Klasse ein/e Proband/in zugeordnet wurde. Mit einem Blick in den Datensatz können Sie nachvollziehen, dass Personen, die der gleichen Subklasse zugeordnet wurden (z.B. Person 1 und Person 4), identische Werte auf den Variablen `sess` und `addic` haben.

3.1 Imputation der *Potential Outcomes*

Das Ziel des Matchings ist es, das *Counterfactual Outcome* (Wert in der nicht-beobachteten Bedingung) zu schätzen.

$$\widehat{Y}_i^0 = \begin{Bmatrix} Y_i & \text{if} & X_i = 0 \\ \frac{1}{M} \sum_{j \in J_{M(i)}} Y_j & \text{if} & X_i = 1 \end{Bmatrix}$$

$$\widehat{Y}_i^1 = \begin{Bmatrix} \frac{1}{M} \sum_{j \in J_{M(i)}} Y_j & \text{if} & X_i = 0 \\ Y_i & \text{if} & X_i = 1 \end{Bmatrix}$$

wobei J die Menge der M gematchten Personen für Person i ist.

Beim exakten Matching werden die nicht-beobachteten Potential Outcomes durch den Mittelwert der Outcomes der Personen in der gleichen Subklasse aber anderen Treatmentbedingung geschätzt/imputiert. Für eine Person in der Interventionsgruppe wird beispielsweise das Potential Outcome für die Kontrollbedingung durch den mittleren Wert der beobachteten Outcomes der KontrollprobandInnen geschätzt, die der gleichen Subklasse wie diese Person angehören.

Wir müssen im Folgenden den Datensatz (`exact_data`) um zwei Variablen ergänzen, um den durchschnittlichen kausalen Effekt schätzen zu können: Für jede Person muss (in der Bedingung, in der sie nicht ist) ein Potential (*Counterfactual*) Outcome bestimmt und im Datensatz hinzugefügt werden. Außerdem werden wir für jede Person, in der Bedingung, in der sie ist, den Mittelwert im Outcome über alle Personen, die in der gleichen Subklasse sind, bestimmen und im Datensatz eintragen. Dies vereinfacht die weitere Berechnung. Für alle Personen kann dann die Differenz aus diesen beiden Outcomes bestimmt werden und der Mittelwert ist der Schätzer für den durchschnittlichen kausalen Effekt:

$$\widehat{ATE} = \frac{1}{N}\sum_{i=1}^{N}(Y_i^1 - Y_i^0)$$

In einem ersten Schritt erstellen wir subklassen- und treatmentspezifische Mittelwerte der Outcome- Variablen ($\frac{1}{M}\sum_{j\in J_{M(i)}} Y_j$), also je einen Mittelwert für alle Kombinationen aus Subklassen und Treatmentbedingungen (also insgesamt 42*2 Mittelwerte). Wir tun dies mithilfe des `aggregate()`-Befehls, mit dem wir in jeder Subklasse die Werte der Outcome-Variable `bdi2` separat für die Personen in der Experimental- und für die Personen in der Kontrollgruppe mitteln.

```
agg <- aggregate(bdi2 ~ subclass + group,
                 data = exakt_data, mean)
head(agg) # Mit head() ruft man nur den oberen Abschnitt eines Objekts auf.
```

```
  subclass group     bdi2
1        1     0 23.00000
2        2     0 28.00000
3        3     0 20.00000
4        4     0 28.33333
5        5     0 19.00000
6        6     0 17.66667
```

Im nächsten Schritt speichern wir die Potential Outcomes der Subklassen in zwei separaten Variablen (Mittelwerte nach Gruppen in Variable getrennt). Das heißt, wir erzeugen

keine neue Information, sondern möchten den Datensatz `agg` so umstrukturieren, dass die Subklassen-Mittelwerte für die Kontrollbedingung ($\hat{Y}_i^0$) und die Subklassen-Mittelwerte für die Treatmentbedingung ($\hat{Y}_i^1$) in zwei verschiedenen Spalten (Variablen) dargestellt werden.

```
wide <- reshape(agg, direction = "wide", sep = "_",
                idvar = "subclass",   # eine Zeile pro subclass
                timevar = "group")    # eine Variable pro Bedingung / Gruppe
head(wide)
```

```
  subclass   bdi2_0 bdi2_1
1        1 23.00000   29.0
2        2 28.00000   17.4
3        3 20.00000   25.0
4        4 28.33333   33.0
5        5 19.00000   23.5
6        6 17.66667   17.0
```

Im erzeugten Datensatz gibt es nun zwei Spalten für die Variable `bdi2`, eine für die mit den Subklassen-Mittelwerten der Kontrollgruppe (`bdi2_0`) und eine mit den Subklassen-Mittelwerten für die Treatmentgruppe (`bdi2_1`).

Der Datensatz `wide` enthält pro Subklasse eine Zeile mit den Subklassen-Mittelwerten für die Kontroll- und die Treatmentgruppe (Potential Outcomes). Dieser Datensatz muss nun mit den Originaldaten zusammengefügt werden, sodass wir wieder eine Zeile pro Person erhalten. Für jede Person soll die neu gewonnene Information über die Potential Outcomes im Datensatz ergänzt werden. Diese Information kann anhand der Subklasse, in der die Person ist, zugeordnet werden. Ganz praktisch gesprochen: Die ersten vier Personen im Datensatz der gematchten Paare (`exact_data`) sind in Subklasse 1. Für sie soll also im Datensatz `exact_data` auf der Variable `bdi2_0` der Wert 23 eingetragen werden und auf der Variable `bdi2_1` der Wert 29. Dies sind die Potential Outcomes für die Subklasse 1, die wir gerade bestimmt haben.

Wir verwenden dafür den `merge()`-Befehl und geben dort die beiden Datensätze an, die wir zusammenfügen möchten und definieren mit dem Argument `by` die Variable, die in beiden Datensätzen vorkommt und anhand derer die Fälle zusammengeführt werden sollen.

```
exakt_data_PO <- merge(exakt_data, wide, by = "subclass") # Vars hinzufügen
head(exakt_data_PO)
```

```
  subclass bdi1 bdi2 age sess cogn addic group  weights bdi2_0   bdi2_1
1        1   43   27  28   24   94     0     1 1.000000   23.0 29.00000
2        1    9   24  33   24   89     0     0 1.676471   23.0 29.00000
3        1   37   22  39   24  102     0     0 1.676471   23.0 29.00000
4        1   27   31  46   24   74     0     1 1.000000   23.0 29.00000
5       10   29   16  32   15   83     0     1 1.000000   40.5 17.66667
6       10   29   41  33   15   79     0     0 2.514706   40.5 17.66667
```

Wir haben den Datensatz jetzt so erstellt, dass wir ihn als nächstes zur Effektschätzung verwenden könnnen.

3.2 Effektschätzung

- Average Treatment Effect: $\widehat{ATE} = \frac{1}{N}\sum_{i=1}^{N}(Y_i^1 - Y_i^0)$
- Wir bilden im erzeugten Datensatz also den Mittelwert über die Differenz der zuvor erzeugten Variablen `bdi2_1` und `bdi2_0`.

```
mean(exakt_data_PO$bdi2_1 - exakt_data_PO$bdi2_0)
```

```
[1] -2.083009
```

- $\widehat{ATE} = -2,08$
- Average Treatment Effect of the Treated: $\widehat{ATT} = \frac{1}{N_T}\sum_{i\in t}(Y_i^1 - Y_i^0)$
- Um diesen Effekt zu bestimmen, bilden wir einen Subdatensatz, der nur die Personen in der Treatmentgruppe beinhaltet, und bilden dann den Mittelwert in gleicher Weise wie zuvor.

```
tg <- exakt_data_PO[exakt_data_PO$group == 1, ]  # Teildatensatz der Treatmentgruppe
mean(tg$bdi2_1 - tg$bdi2_0)
```

```
[1] -3.205994
```

- $\widehat{ATT} = -3,21$

3.3 Problem beim Exakten Matching

Bei steigender Anzahl der Kovariaten wird exaktes Matching oft schwierig:

```r
exakt2 <- matchit(group ~ age + sess + cogn + addic,
                  data = bdi_data, method = "exact")
exakt2
```

```
A matchit object
 - method: Exact matching
 - number of obs.: 255 (original), 2 (matched)
 - target estimand: ATT
 - covariates: age, sess, cogn, addic
```

Bei Hinzunahme aller Kovariaten kann mit exaktem Matching nur eine einzige Person aus der Treatmentgruppe mit einer Person aus der Kontrollgruppe gematched werden.

4 Nearest Neighbor Matching

4.1 Entscheidungen

- Distanzmetrik
  - Bestimmung der Ähnlichkeit zweier Personen auf Basis der Kovariaten
- Matching Strategie
  - Anzahl der Matches (1:1, 1:M)
  - Toleranz in Bezug auf Ähnlichkeit (Caliper)
  - Mit oder ohne Zurücklegen (mehrfache Verwendung derselben Personen)
- Algorithmus
  - Kriterium und Reihenfolge (Greedy, Optimal, Nearest Neighbor, …)
- Indikatoren gelungenen Matchings
  - Reduzierung der Unterschiede zwischen Gruppen auf den Kovariaten
  - Gruppengrößen (Ausmaß des Datenverlusts)

4.2 Nearest Neighbor Matching

- Nearest Neighbor Matching mit `matchit()` wird spezifiziert über das Argument `method="nearest"`. Als Distanzmaß wählen wir die Mahalanobis-Distanz mit `distance = "mahalanobis"`. Wir werden nun alle vorhandenen Kovariaten im Modell berücksichtigen.
- Da im Nearest Neighbor Matching der `matchit()`-Befehl einen Zufallsprozess integriert hat, werden wir als erstes alle den gleichen Startwert definieren. Das führt dazu, dass wir alle das gleiche Ergebnis in den weiteren Analysen erhalten und dieses Ergebnis reproduzieren können, wenn wir jedes Mal vor Anwendung des `matchit()`-Befehls wieder den gleichen Startwert definieren. Diesen Startwert können wir beliebig wählen, es sollte nur bei allen derselbe Wert sein. Wir wählen hier nun den Startwert 1 und definieren ihn in R mit dem Befehl `set.seed()`.

```r
set.seed(1)
near <- matchit(group ~ age + sess + cogn + addic + bdi1,
                data = bdi_data,
                method = "nearest",
                distance = "mahalanobis",  # Mahalanobis-Distanz als Distanzmetrik
                ratio = 1,           # 1:1 Matching (default)
                replace = FALSE)     # ohne Zurücklegen
```

Nearest Neighbor Matching führt per Voreinstellung einen *greedy matching* Algorithmus durch. Alternativ kann auch Optimal Matching benutzt werden, bei dem ein globales Distanzmaß minimiert wird (minimale durchschnittliche absolute Distanz über alle Paare). Dafür verwendet man im `matchit()`-Befehl die Spezifikation `method = "optimal"`.

Über das Argument `ratio` wird gesteuert, wieviele Kontrollprobanden einem Treatmentprobanden zugeordnet werden. Über das Argument `replace` wird eingestellt, ob matching mit oder ohne Zurücklegen stattfindet.

Übersicht über die Matching-Ergebnisse und Verteilung der Kovariaten in den Gruppen vor und nach dem Matching bekommt man mithilfe des `summary()`-Befehls. (Sie können auch schon den `plot()`-Befehl ausprobieren. Diesen stellen wir in den nächsten Skripten ausführlicher dar.)

```r
summary(near,
        standardize = TRUE,
        improvement = TRUE)  # auch: plot(near)
```

```
Call:
```

```
matchit(formula = group ~ age + sess + cogn + addic + bdi1, data = bdi_data,
    method = "nearest", distance = "mahalanobis", replace = FALSE,
    ratio = 1)

Summary of Balance for All Data:
      Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean eCDF Max
age         35.8158       33.8324          0.2849     0.8527    0.0580   0.1154
sess        20.2763       15.3631          0.4761     0.7064    0.1107   0.2568
cogn        82.7368       91.9609         -0.8903     0.4451    0.1415   0.3267
addic        0.4211        0.4246         -0.0071          .    0.0035   0.0035
bdi1        30.9605       20.9553          1.0375     0.6310    0.1916   0.4072

Summary of Balance for Matched Data:
      Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean eCDF Max
age         35.8158       34.8421          0.1399     1.3770    0.0376   0.1184
sess        20.2763       17.7237          0.2474     0.8985    0.0569   0.1184
cogn        82.7368       86.8026         -0.3924     0.7841    0.0672   0.2632
addic        0.4211        0.4211          0.0000          .    0.0000   0.0000
bdi1        30.9605       28.0526          0.3015     0.8588    0.0572   0.1974
      Std. Pair Dist.
age            0.4536
sess           0.4335
cogn           0.6540
addic          0.0000
bdi1           0.5389

Percent Balance Improvement:
      Std. Mean Diff. Var. Ratio eCDF Mean eCDF Max
age              50.9     -100.7      35.1     -2.6
sess             48.0       69.2      48.6     53.9
cogn             55.9       69.9      52.5     19.4
addic           100.0          .     100.0    100.0
bdi1             70.9       66.9      70.2     51.5

Sample Sizes:
          Control Treated
All           179      76
Matched        76      76
Unmatched     103       0
Discarded       0       0
```

Wir sehen, dass sich die Verteilungen nach dem Matching deutlich ähnlicher sind als zuvor. Zum Beispiel haben sich die `Means Treated` und `Means Control` nach dem Matching angenähert, was in einer verringerten Mittelwertsdifferenz (`Std. Mean Diff.`) zum Ausdruck kommt. Als Cut-off-Wert für einen gelungenen Matchingprozess wird ein maximaler standardisierter Bias von 0.25 (oder sogar von 0.10) verwendet (siehe auch nächste Sitzung zu den Propensity Scores). Selbst den weniger strengen Cut-off von 0.25 überschreiten wir hier sowohl für die Kovariate `cogn` als auch für `bdi1`. Idealerweise bräuchten wir also eine größere Stichprobe in der Kontrollgruppe, um passende MatchingpartnerInnen zu finden.

Im Abschnitt `Percent Balance Improvement` kann in der ersten Spalte (`Std. Mean Diff`) abgelesen werden, um wie viel Prozent die Unterschiede zwischen den beiden Gruppen auf der jeweiligen Kovariaten reduziert werden konnten.

Beim Nearest Neighbor Matching mit `ratio = 1` wird jeder_m ProbandIn in der Treatment-Gruppe ein_e ProbandIn aus der Kontrollgruppe zugeordnet. Daher gibt es beim Nearest Neighbor Matching keine `subclass` Variable. Informationen über die gebildeten Paare können abgerufen werden über

```
near$match.matrix
```

```
head(near$match.matrix)
```

Die Zeilennummern entsprechen den Zeilennummern der ProbandInnen in der Treatmentgruppe im ursprünglichen Datensatz (hier: `bdi_data`), die Werte in Anführungszeichen entsprechen den Zeilennummern der gematchten KontrollprobandInnen im ursprünglichen Datensatz.

4.3 Effektschätzung

Wir extrahieren zunächst den Datensatz der gematchten Personen mithilfe des `match.data()`-Befehls:

```
near_data <- match.data(near)
```

Wenn beim Nearest Neighbour Matching jeder_m Treatmentgruppen-ProbandIn exakt gleich viele KontrollprobandInnen zugeordnet werden und das Matching ohne Zurücklegen (`replace=FALSE`) durchgeführt wurde, ist eine Gewichtung (also ein Mitteln über alle Personen in der Kontrollgruppe, die derselben Person in der Treatmentgruppe zugeordet wurden, wie wir es oben beim exakten Matching getan haben) nicht notwendig. Stattdessen

kann der $\widehat{ATE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y}_i^1 - \hat{Y}_i^0)$ dann geschätzt werden als Differenz der gruppenspezifischen Mittelwerte (also als $\widehat{ATE} = (\frac{1}{N}\sum_{i=1}^{N}\hat{Y}_i^1) - (\frac{1}{N}\sum_{i=1}^{N}\hat{Y}_i^0)$). Wir bestimmen diese Differenz in einem einfachen Regressionsmodell mit `bdi2` als abhängiger Variable und der Gruppe (`group`) als Prädiktor. Dabei müssen wir darauf achten, den Datensatz nach dem Matching (`near_data`) zu verwenden.

```r
# Effektschätzung
lm(bdi2 ~ group, data = near_data)
```

```
Call:
lm(formula = bdi2 ~ group, data = near_data)

Coefficients:
(Intercept)        group
     29.803       -6.763
```

- $\widehat{ATE}$ = -6.76: Der durchschnittliche kausale Effekt für die achtsamkeitsbasierte Intervention (als Ergänzung zur Standard-Therapie) ist eine Verminderung des Depressionswertes um 6.76 Punkte.
- Vorausgesetzt jeder_m TreatmentprobandIn werden gleich viele Kontrollproband*innen zugeordnet, entspricht beim Nearest Neighbor Matching der $\widehat{ATE}$ dem $\widehat{ATT}$.