Appendix B
Introduction to R Code Implementation

This appendix includes a brief introduction to the implementation of super learning and the TMLE in R. Packages and supplementary code are posted online at http://www.targetedlearningbook.com. We conclude with a few coding guides for data structures and research questions presented in Parts II–IX. The book's Web site will be a continually updated resource for new code, demonstrations, and packages.

Suppose you want to implement super learning. In order to include user-specified regressions in parametric statistical models (default specification is main terms), you must write additional wrappers for the super learner function. A sample wrapper is included below, and more information on writing wrappers is on our Web site.

```
SL.glm.2 <- function(Y.temp, X.temp, newX.temp,
      family, obsWeights, ...){
fit.glm <- glm(Y.temp ~ A+W1, data=X.temp,
      family=family, weights = obsWeights)
out <- predict(fit.glm, newdata=newX.temp,
      type="response")
fit <- list(object=fit.glm)
foo <- list(out=out, fit=fit)
class(foo$fit) <- c("SL.glm")
return(foo)}
```

Sample code to run the SuperLearner package in a simulated data set called "samp," including a small proposed collection of algorithms, is included below. (Package author: Eric C. Polley.)

```
library(SuperLearner)
SL.library <- list("SL.glm.2", "SL.bayesglm",
      "SL.glmnet", "SL.glmnet.alpha50", "SL.gam",
      "SL.gam.3", "SL.nnet", "SL.nnet.4")
fit.SL <- SuperLearner(Y=samp$Y,X=samp[,c(1:3,5:6)],
      SL.library=SL.library, family=binomial,
      method="NNLS", verbose=TRUE, V=10)
```

M.J. van der Laan and S. Rose, *Targeted Learning: Causal Inference for Observational and Experimental Data*, Springer Series in Statistics, DOI 10.1007/978-1-4419-9782-1_31,

Note that there are several options available beyond the selection of the algorithms in the super learner function. Familiarize yourself with these options in the help file. In order to obtain the *cross-validated* risk of the super learner, one must run CV.SuperLearner. Sample code is given below.

```
fit.SL.CV <- CV.SuperLearner(Y=samp$Y,
      X=samp[,c(1:3,5:6)],SL.library=SL.library,
      family=binomial,method="NNLS",
      verbose=TRUE, outside.V=10, inside.V=10)
QSL.risk <- mean((samp$Y-fit.SL.CV$pred.SL)^2)
```

Now we give sample code to implement the R package tmle (Gruber and van der Laan 2011) to estimate the parameter $\Psi(P_0) = E_{W,0}[E_0(Y \mid A = 1, W) - E_0(Y \mid A = 0, W)]$ in simulated data. We refer interested readers to Gruber and van der Laan (2011) and the supplemental materials on our Web site for additional code, explanations, and options.

```
run1 <- tmle(Y, A, W, family="binomial",
      Q.SL.library = c("SL.glm", "SL.step",
      "SL.DSA.2"), gform = A ~ W1 + W2 + W3,)
```

Implementation notes for other data structures and parameters of interest presented throughout the book for selected chapters are given below. This is not a comprehensive tutorial. As noted, additional materials are available online.

Chapters 13–15: Case-control implementation. Implementation of super learning and TMLE for case-control studies and other biased study designs is straightforward. Using weights requires a simple weight statement within the functions, for example, in Chapter 15, we call

```
fit.SL <- SuperLearner(Y=ccData$death_yn,
      X=ccData[,2:167],SL.library=SL.library,
      family=binomial,method="NNLS",obsWeights=weight,
      verbose=TRUE, V=10)
```

when generating the function for risk score using super learner.

Chapter 16: Super learning for censored data structures. Required wrappers to run super learning in censored data structures are available on our Web site. Below we present an example of hazard estimation in the publicly available lung cancer data set studied in Chapter 16 using these wrappers and the collection of algorithms discussed in the chapter. (Author: Eric C. Polley.)

```
data(lung)
subLung <- subset(lung, select = c(time, status, age,
      ph.ecog, ph.karno, pat.karno))
subLung$female <- (lung$sex - 1)
subLung <- subLung[complete.cases(subLung), ]
```

```
## Expand subLung to Long Format
longData <- SuperLearner:::createDiscrete(time =
    subLung$time, event = (subLung$status == 2),
    dataX = subset(subLung, select =
    -c(time, status)), n.delta = 30)

## Super Learner
fit.SL <- SuperLearner(Y = longData$N.delta,
    X = data.frame(age = longData$age, ph.ecog =
    longData$ph.ecog, female = longData$female,
    ph.karno = longData$ph.karno, pat.karno =
    longData$pat.karno, time = longData$delta.upper),
    V = 10, SL.library = SL.library, shuffle = FALSE,
    verbose = TRUE, id = longData$ID, family =
    binomial(), method = "NNLS")

## CV Super Learner
fit.SL.CV <- CV.SuperLearner(Y = longData$N.delta,
    X = data.frame(age = longData$age, ph.ecog =
    longData$ph.ecog, female = longData$female,
    ph.karno = longData$ph.karno, pat.karno =
    longData$pat.karno, time = longData$delta.upper),
    outside.V = 10, inside.V = 10, SL.library =
    SL.library, shuffle = FALSE, verbose = TRUE, id =
    longData$ID, family = binomial(), method = "NNLS")
```

Chapters 19–21: C-TMLE. A simple example using the C-TMLE code available on our Web site is included below. (Author: Susan Gruber.)

```
set.seed(10)
n <-1000
W <- matrix(rnorm(n*5), ncol=5)
colnames(W) <- paste("W",1:5,sep="")
logitA <- .3*W[,1]+.2*W[,2]-3*W[,3]
pA <- plogis(logitA)
Wstar <- rbinom(n,1,pA)
pA3 <- plogis(.15*logitA)
A3 <- rbinom(n,1,pA3)
Y3 <- A3+.5*W[,1]-8*W[,2]+W[,3]+8*W[,3]^2
      -2*W[,5] +rnorm(n)
d.sim3 <- data.frame(Y=Y3, A=A3, W, Wstar)
ctmle.sim3 <- ctmle(Y=d.sim3$Y, A=d.sim3$A,
      W=d.sim3[,-c(1:2)])
summary(ctmle.sim3)
```

Chapter 25: Longitudinal data. We include code that uses the DAIFI data and related functions (available on our Web site) to run the analysis presented in the corresponding chapter. (Author: Antoine Chambaz.)

```
source("DAIFI.extract_fun.R")
true <- 0.652187 ## based on 1e6 simulated data
obs <- getSample(3001, FALSE)
fit0 <- getInitialFit(obs, whole = TRUE, V = 10)
gcomp <- getEstimate(obs, fit0)
gcomp <- gcomp[1:4]
fit1 <- updateFit(obs, fit0)
tmle <- getEstimate(obs, fit1)
tmle <- tmle[1:4]
```

Chapter 29: Bayesian TMLE. We include code to run an example of Bayesian targeted learning. A uniform prior on the interval $(-1, 1)$ is used, logistic regressions are used as initial estimators of $\bar{Q}_0(A, W)$ and $g_0(A \mid W)$, and 1000 posterior observations are drawn by using the function BTL. The simulated data and required functions are available on our Web site. (Author: Iván Díaz Muñoz.)

```
data <- read.csv("data.csv")
prior.psi <- list(func = debeta, args = list(a = -1,
      b = 1, shape1 = 1, shape2 = 1))
Y <- data$Y
A <- data$A
W <- data.frame(W1 = data$W1, W2 = data$W2)
Q <- Y ~ A*W1*W2
g_A <- A ~ W1*W2
family <- "binomial"
output <- BTL(Y, A, W, prior.psi , Q, g_A, family)
```