Code

TMLE step by step

By: Miguel Angel Luque Fernandez miguel-angel.luque@lshtm.ac.uk (mailto:miguel-angel.luque@lshtm.ac.uk)

October 25th, 2016

1 Introduction

During the last 30 years, the modern epidemiology has been able to identify some important drawbacks of the classic epidemiologic methods (bivariate or multivariate adjusted models) when the focus is to explanation the main effect of a risk factor on a disease or outcome.

Causal Inference, first introudced in Social Science by Donal Rubin (Rubin, 1974) and later in Epidemiology and Biostatistics by James Robins (Greenland and Robins, 1986), the **Neyma-Rubin Potential Outcomes framework** (Rubin, 1974),(Rubin, 2011) has provided the theory and statistical methods needed to identify and overcome recurrent problems in observational epidemologic research, such as:

1. non collapsibility of the odds and hazard ratios,
2. impact of paradoxical effects due to conditioning on colliders,
3. left truncation,
4. prevalent cases,
5. selection bias related with the vague understanding of the effect of time on exposure and outcome and,
6. effect of time dependent confounding and mediators,
7. etc.

To control for confounding, the classical epidemilogic methods require making the assumption that the effect measure is constant across levels of confounders included in the model.

Alternatively, James Robins in 1986 demonstrated that using standardization, implemented through the use of the **G-formula**, allowed to obtain unconfounded marginal estimation of the causal average treatment effect (ATE) under causal nontestable assumptions (Greenland and Robins, 1986), (Robins *et al.*, 2000). The most commonly used estimator for a binary treatment effect is the risk difference or **ATE** = $\psi(P_0)$.

2 The G-Formula

$$\psi(P_0) = \sum_w \left[\sum_y P(Y = y \mid A = 1, W = w) - \sum_y P(Y = y \mid A = 0, W = w) \right] P(W = w)$$

where,

$$P(Y = y \mid A = a, W = w) = \frac{P(W = w, A = a, Y = y)}{\sum_y P(W = w, A = a, Y = y)}$$

is the conditional probability distribution of Y = y, given A = a, W = w and,

$$P(W = w) = \sum_{y,a} P(W = w, A = a, Y = y)$$

- Classical epidemilogic methods require making the assumption that the effect measure is constant across levels of confounders included in the model. However, **Standardization** allows us to obtain an unconfounded summary effect measure without requiring this assumption. The **G-formula** is a *generalization of standardization* (Greenland and Robins, 1986).
- The ATE can be estimated **non-parametrically** using the G-formula. However, the course of dimensionality in observational studies limits its estimation.
- Hence, the estimation of the ATE using the G-formula relies mostly on **parametric modelling** assumptions and maximum likelihood estimation. The **correct model specification** in parametric modelling is crucial to obtain unbiased estimates of the true ATE (Rubin, 2011).

However, Mark van der Laan and collaborators have developed a double-robust estimation procedure **to reduce bias against misspecification**. The targeted maximum likelihood estimation (TMLE) is a semiparametric, efficient substitution estimator (Laan and Rose, 2011).

3 TMLE

TMLE allows for data-adaptive estimation while obtaining valid statistical inferencebased on the targeted minimum loss-based estimation and machine learning algorithms to minimize the risk of model misspecification (Laan and Rose, 2011). The main characteristics of **TMLE** are:

1. **TMLE** is a general algorithm for the construction of double-robust, semiparametric, efficient substitution estimators. **TMLE** allows for data-adaptive estimation while obtaining valid statistical inference.

2. **TMLE** implemtation uses the G-computation estimand (G-formula). Briefly, the **TMLE** algorithm uses information in the estimated exposure mechanism P(A|W) to update the initial estimator of the conditional expectaction of the outcome given the treatment and the set of covariates W, E_0(Y|A,W).

3. The targeted estimates are then substituted into the parameter mapping Ψ. The updating step achieves a targeted bias reduction for the parameter of interest $\psi(P_0)$ (the true target parameter) and serves to solve the efficient score equation, namelly Influence Curve (IC). As a result, **TMLE** is a double robust estimator.

4. **TMLE** it will be consistent for $\psi(P_0)$ if either the conditional expectation E_0(Y|A,W) or the exposure mechanism P_0(A|W) are estimated consistently.

5. **TMLE** will be efficient if the previous two functions are consistently estimated achieveing the lowest asymptotic variance among a large class of estimators. These asymptotic properties typically translate into lower bias and variance in finite samples (Bühlmann *et al.*, 2016).

6. The general formula to estimate the ATE using the TMLE method:

$$\psi TMLE, n = \Psi(Q_n^*) = \frac{1}{n}\sum_{i=1}^{n} \bar{Q}_n^1\,(1,\ W_i) - \bar{Q}_n^1\,(0,\ W_i).\ (1)$$

7. The efficient influcence curve (IC) based on Hampel seminal paper (Hampel, 1974) is applied for statistical inference using TMLE:

$$IC_n(O_i) = \left(\frac{I\,(A_i = 1)}{g_n\,(1\,|W_i)} - \frac{I\,(A_i = 0)}{g_n\,(0\,|W_i)}\right)\left[Y_i - \bar{Q}_n^1\,(A_i, W_i)\right] + \bar{Q}_n^1\,(1,\ W_i) - \bar{Q}_n^1\,(0,\ W_i) - \psi TMLE, n.\ (2)$$

where the variance of the ATE:

$$\sigma(\psi_0) = \sqrt{\frac{Var(IC_n)}{n}}.\ (3)$$

8. The procedure is available with standard software such as the **tmle** package in R (Gruber and Laan, 2011).

4 Structural causal framework

4.1 Direct Acyclic Graph (DAG)

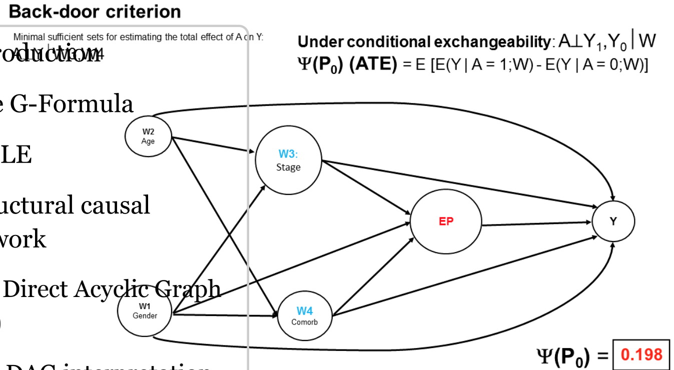

Figure 1. Direct Acyclic Graph
Source: Miguel Angel Luque-Fernandez

4.2 DAG interpretation

The ATE is interpreted as the population risk difference in one-year mortality for lung cancer patients diagnosed via emergency presentations versus non-emergency presentations. Under causal assumptions, and compared with non-emergency presentations of lung cancer, the risk difference of one-year mortality for emregency presentations increases by approximately 20%.

5 Causal assumptions

Under the counterfactual framework the following assumptions have to be considered to estimate the $\psi(P_0)$ (ATE) with a mondel for P_0 augmented with additional nontestatble cuasal assumptions (Rubin, 2011), (Laan and Rose, 2011):

5.1 CMI or Randomization

$(Y_0, Y_1 \perp A|W)$ of the binary treatment effect (A) on the outcome (Y) given the set of observed covariates (W), where W = (W1, W2, W3, … , Wk).

5.2 Positivity

a ϵ A: P(A=a | W) > 0
P(A=1|W=w) > 0 and P(A=0| W = w) > 0 for each possible w.

5.3 Consistency or SUTVA:

The observed outcome value, under the observed treatment, is equal to the counterfactual outcome corresponding to the observed treatment for identical independent distributed (i.i.d.)

variables.

6 TMLE flow chart

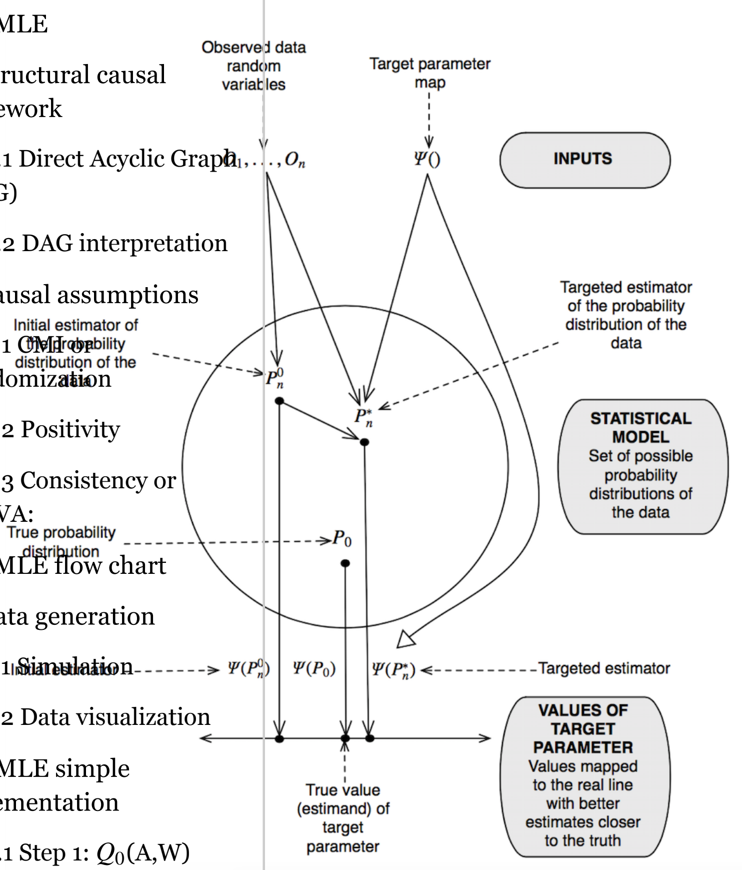

Figure 2. TMLE flow chart (Road map)
Source: Mark van der Laan and Sherri Rose. Targeted learning: causal inference for observational and experimental dataSpringer Series in Statistics, 2011.

7 Data generation

7.1 Simulation

In R we create a function to generate the data. The function will have as input **number of draws** and as output the generated **observed data** (ObsData) including the counterfactuals (Y1, Y0).

The simulated data replicationg the DAG in Figure 1:

1. Y: mortality binary indicator (1 death, 0 alive)
2. A: binary treatment for emergency presentation at cancer diagnosis (1 EP, 0 NonEP)
3. W1: Gender (1 male; 0 female)
4. W2: Age at diagnosis (0 <65; 1 >=65)

5. W3: Cancer TNM classification (scale from 1 to 4)
6. W4: Comorbidities (scale from 1 to 5)

Hide

```
#install.packages("broom")
options(digits=4)
generateData <- function(n){
  w1 <- rbinom(n, size=1, prob=0.5)
  w2 <- rbinom(n, size=1, prob=0.65)
  w3 <- round(runif(n, min=0, max=4), digits=3)
  w4 <- round(runif(n, min=0, max=5), digits=3)
  A  <- rbinom(n, size=1, prob= plogis(-0.4 + 0.2*w2 + 0.15*w3 + 0.2*w4 + 0.15*w2*w4))
  Y  <- rbinom(n, size=1, prob= plogis(-1 + A -0.1*w1 + 0.3*w2 + 0.25*w3 + 0.2*w4 + 0.15*w2*w4))

  # counterfactual
  Y.1 <- rbinom(n, size=1, prob= plogis(-1 + 1 -0.1*w1 + 0.3*w2 + 0.25*w3 + 0.2*w4 + 0.15*w2*w4))
  Y.0 <- rbinom(n, size=1, prob= plogis(-1 + 0 -0.1*w1 + 0.3*w2 + 0.25*w3 + 0.2*w4 + 0.15*w2*w4))

  # return data.frame
  data.frame(w1, w2, w3, w4, A, Y, Y.1, Y.0)
}
set.seed(7777)
ObsData <- generateData(n=10000)
True_Psi <- mean(ObsData$Y.1-ObsData$Y.0);
cat(" True_Psi:", True_Psi)
```

```
 True_Psi: 0.198
```

Hide

```
Bias_Psi <- lm(data=ObsData, Y~ A)
cat("\n")
```

Hide

```
cat("\n Naive_Biased_Psi:",summary(Bias_Psi)$coef[2, 1])
```

```
 Naive_Biased_Psi: 0.2631
```

Hide

```
Naive_Bias <- ((summary(Bias_Psi)$coef[2, 1])-True_Psi); cat("\n Naives bias:", Naive_Bias)
```

```
Naives bias: 0.06509
```

Hide

```
Naive_Relative_Bias <- (((summary(Bias_Psi)$coef[
2, 1])-True_Psi)/True_Psi)*100; cat("\n Relative
Naives bias:", Naive_Relative_Bias,"%")
```

```
Relative Naives bias: 32.88 %
```

7.2 Data visualization

Hide

```
# DT table = interactive
# install.packages("DT") # install DT first
library(DT)
datatable(head(ObsData, n = nrow(ObsData)), optio
ns = list(pageLength = 5, digits = 2))
```

Show 5 **entries** **Search:**

	w1	w2	w3	w4	A	Y	Y.1	Y.0
1	0	1	3.282	3.541	1	1	0	0
2	1	0	0.047	1.425	1	1	0	0
3	1	0	3.354	4.158	1	1	1	0
4	1	1	2.085	1.737	1	1	0	0
5	0	1	2.487	0.419	1	1	0	1

Showing 1 to 5 of 10,000 entries

Previous 1 2 3 4 5 … 2000 Next

8 TMLE simple implementation

8.1 Step 1: Q_0(A,W)

Estimation of the initial probability of the outcome (Y) given the treatment (A) and the set of covariates (W), denoted as $Q_0(\text{A},\mathbf{W})$.

To estimate Q_0(A,**W**) we can use a standard logistic regression model:

$$\text{logit}[P(Y = 1|A, W)] = \beta_0 + \beta_1 A + \hat{\beta}_2^T W.$$

Therefore, we can estimate the initial probability as follows:

$$\bar{Q}^0(A, W) = \text{expit}(\hat{\beta}_0 + \hat{\beta}_1 A + \hat{\beta}_2^T W).$$

The predicted probability can be estimated using the Super-Learner library implemented in the R package "Super-Learner" (Van der Laan *et al.*, 2007) to include any terms that are functions of A or W (e.g., polynomial terms of A and W, as well as the interaction terms of A and W, can be considered).

Consequently, for each subject, the predicted probabilities for both potential outcomes $\bar{Q}^0(0, W)$ and $\bar{Q}^0(1, W)$ can be estimated by setting A = 0 and A = 1 for everyone respectively:

$$\bar{Q}^0(0, W) = \text{expit}(\hat{\beta}_0 + \hat{\beta}_2^T W),$$

and,

$$\bar{Q}^0(1, W) = \text{expit}(\hat{\beta}_0 + \hat{\beta}_1 A + \hat{\beta}_2^T W).$$

Note: see appendix one for a short introduction to the Super-Learner and ensemble learning thecniques.

Hide

```
ObsData <-subset(ObsData, select=c(w1,w2,w3,w4,A,Y))
Y  <- ObsData$Y
A  <- ObsData$A
w1 <- ObsData$w1
w2 <- ObsData$w2
w3 <- ObsData$w3
w4 <- ObsData$w4
m <- glm(Y ~ A + w1 + w2 + w3 + w4, family=binomial, data=ObsData)
Q  <- cbind(QAW = predict(m),
            Q1W = predict(m, newdata=data.frame(A = 1, w1, w2, w3, w4), type="response"),
            Q0W = predict(m, newdata=data.frame(A = 0, w1, w2, w3, w4), type="response"))
Q0 <- as.data.frame(Q);mean(Q0$Q1W-Q0$Q0W)
```

```
[1] 0.1992
```

8.2 Step 2: $g_0(A, W)$

Estimation of the probability of the treatment (A) given the set of covariates (W), denoted as $g_0(A, W)$. We can use again a logistic regression model and to improve the prediction algorithm we can use the Super-Learner library or any other machine learning estrategy:

$$\text{logit}[P(A = 1|W)] = \beta_0 + \beta_1^T W.$$

Then, we estimate the predicted probability of P(A|W) = $\hat{g}(1, W)$ using:

$$\hat{g}(1, W) = \text{expit} = (\hat{\beta}_0 + \hat{\beta}_2^T W).$$

Hide

```
g <- glm(A ~ w2 + w3 + w4, family = binomial)
g1w = predict(g, type ="response");summary(g1w)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.358   0.594   0.681   0.671   0.759   0.875 
```

8.3 Step 3: HAW and ϵ

This step aims to find a better prediction model targeted at minimising the mean squared error (MSE) for the potential outcomes by using the so-called efficient IC estimation equation. For the ATE on step convergence is guaranteed given $\bar{Q}^0$ and $\hat{g}(1, W)$.

The fluctuating parameter is modelled using a parametric working model to estimate the fluctuation parameters ϵ_0 and ϵ_1 as follows:

$$\bar{Q}^1(A, W) = \text{expit}\left[\text{logit}\left(\bar{Q}^0(A, W)\right) + \hat{\epsilon}_0 H_0(A, W) + \hat{\epsilon}_1 H_1(A, W)\right] \quad (5)$$

$$\bar{Q}^1(0, W) = \text{expit}\left[\text{logit}\left(\bar{Q}^0(A, W)\right) + \hat{\epsilon}_0 H_0(A, W)\right]$$

$$\bar{Q}^1(1, W) = \text{expit}\left[\text{logit}\left(\bar{Q}^0(A, W)\right) + \hat{\epsilon}_1 H_1(A, W)\right]$$

Where,

$$H_0(A, W) = \frac{I(A = 0)}{\hat{g}(0|W)} \text{ and, } H_1(A, W) = \frac{I(A = 1)}{\hat{g}(1|W)}$$

are referred to as clever covariates (note that $\hat{g}(A|W)$ is estimted from step 2).

The fluctuation parameters $(\hat{\epsilon}_0\ , \ \hat{\epsilon}_1)$ are estimated using maximum likelihood procedures by setting logit($\bar{Q}^0(A, W)$) as an offset in a intercept-free logistic regression with H_0 and H_1 as independent variables.

Afterwards, the estimated probability of the potential outcomes is updated by the substitution parameters ($\hat{\epsilon}_0$, $\hat{\epsilon}_1$). The substitution update is performed by setting A = 0 and A = 1 for each subject in the initial estimate probability of the potential outcomes $\bar{Q}^1(0, W)$, $\bar{Q}^1(1, W)$, as well as in the clever covariates $H_0(0, W)$ and $H_1(1, W)$.

Hide

```
#Model 5: Clever covariate and fluctuating/substitution paramteres
h <- cbind(A/g1w -(1-A)/(1-g1w), 1/g1w, -1/(1-g1w))
epsilon <- coef(glm(Y ~ -1 + h[,1] + offset(Q[,"QAW"]), family = binomial));epsilon
```

```
h[, 1]
0.001189
```

8.4 Step 4: $\bar{Q}_n^*$

For the ATE, the updated estimate of the potential outcomes only needs one iteration $\Psi(\bar{Q}_n^*)$ from $\bar{Q}^0(A, W)$ => $\bar{Q}^1(A, W)$. Therefore, model (5) targets $E[\hat{Y}_{A=0}]$ and $E[\hat{Y}_{A=1}]$ simultaneously by including both $H_0(A, W)$ and $H_1(A, W)$ in the model. Hence ψ is finally estimated as follows:

$$\psi TMLE, n = \Psi(Q_n^*) = \frac{1}{n}\sum_{i=1}^{n} \bar{Q}_n^1(1,\ W_i) - \bar{Q}_n^1(0,\ W_i). \quad (1)$$

Hide

```
Qstar <- plogis(Q + epsilon*h)
Psi <- mean(Qstar[,"Q1W"] - Qstar[,"Q0W"]);cat("TMLE_Psi:", Psi)
```

```
TMLE_Psi: 0.04594
```

Hide

```
cat("\n TMLE.SI_bias:", abs(True_Psi-Psi))
```

```
TMLE.SI_bias: 0.1521
```

Hide

```
cat("\n Relative_TMLE.SI_bias:",abs(True_Psi-Psi)/True_Psi*100,"%")
```

```
 Relative_TMLE.SI_bias: 76.8 %
```

8.5 Step 5: Inference

TMLE uses the efficient influence curve (IC) for inference (i.e., to obtain standard errors for ψ).

$$IC_n(O_i) = \left(\frac{I(A_i = 1)}{g_n(1 \mid W_i)} - \frac{I(A_i = 0)}{g_n(0 \mid W_i)}\right)\left[Y_i - \bar{Q}_n^1(A_i, W_i)\right] + \bar{Q}_n^1(1,\ W_i) - \bar{Q}_n^1(0,\ W_i) - \psi TMLE, n. (2)$$

where the standard deviation for ψ is estimated as follows:

$$\sigma(\psi_0) = \sqrt{\frac{Var(IC_n)}{n}}. (3)$$

Note: see appendix two for a short introduction to the Influence Curve theory.

Hide

```
Q <- as.data.frame(Q)
IC <- h[,1]*(Y-Q$QAW) + Q$Q1W - Q$Q0W - Psi;summary(IC)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 -3.610  -1.550  -0.878  -0.678  -0.073  13.500 
```

Hide

```
n <- nrow(ObsData)
varHat.IC <- var(IC)/n; varHat.IC
```

```
[1] 0.0002032
```

Hide

```
#Psi and 95%CI for Psi
cat("\n TMLE_Psi:", Psi)
```

```
 TMLE_Psi: 0.04594
```

Hide

```
cat("\n 95%CI:", c(Psi-1.96*sqrt(varHat.IC),  Psi +1.96*sqrt(varHat.IC)))
```

```
 95%CI: 0.018 0.07388
```

Hide

```
cat("\n TMLE.SI_bias:", abs(True_Psi-Psi))
```

```
 TMLE.SI_bias: 0.1521
```

Hide

```
cat("\n Relative_TMLE.SI_bias:",abs(True_Psi-Psi)/True_Psi*100,"%")
```

```
 Relative_TMLE.SI_bias: 76.8 %
```

9 TMLE vs. AIPTW

1. The advantages of **TMLE** have been repeatedly demonstrated in both simulation studies and applied analyses (Laan and Rose, 2011).

2. Evidence shows that **TMLE** provides the less unbiased ATE estimate compared with other double-robust estimators (Neugebauer and Laan, 2005), (Laan and Rose, 2011) such as the combination of regression adjustment with inverse probability of treatment weighting (IPTW-RA) and the augmented inverse probability of treatment weighting (AIPTW). The **AIPTW** estimation is a two step procedure with two equations (propensity score equation and, mean outcome equation).

3. To estimate the ATE using the **AIPTW** estimator one can set the estimation equation (EE) (4) equal to cero and use bootstrap to derive 95% confidence intervals (CI). However, solving the EE using the generalized method of moments (GMM), stacking both equations (propensity score and outcome), reduces the estimation and inference steps to only one. However, given that the propensity score in

equation (4) can easily fall outside the range [0, 1] (if for some observations $g_n(1|W_i)$ is close to 1 or 0) the **AIPTW** estimation can be unstable (near violation of the positivity assumption). This represents the price of not being a substitution estimator as **TMLE**.

$$\psi_n^{AIPTW-ATE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{I(A_i=1)}{g_n(1|W_i)} - \frac{I(A_i=0)}{g_n(0|W_i)}\right)\left[Y_i - \bar{Q}_n^0(A_i, W_i)\right] + \frac{1}{n}\sum_{i=1}^{n}\bar{Q}_n^0(1, W_i) - \bar{Q}_n^0(0, W_i) . \quad (4)$$

Hide

```
AIPTW <- mean((h[,1]*(Y-Q$QAW))+(Q$Q1W-Q$Q0W)); A
IPTW
```

```
[1] 0.1948
```

Hide

```
cat("\n AIPTW_bias:", abs(True_Psi-AIPTW))
```

```
 AIPTW_bias: 0.003211
```

Hide

```
cat("\n Relative_AIPTW_bias:",abs(True_Psi-AIPTW)
/True_Psi*100,"%")
```

```
 Relative_AIPTW_bias: 1.622 %
```

Compared with AIPTW, TMLE showed smaller relative bias.

10 TMLE using the Super-Learner

With TMLE we can call the Super-Learner (SL). The SL is a R-package using V-fold cross-validation and ensembled learning (prediction using the all the predictions of multiple stacked learning algorithms) techniques to improve model prediction performance (Breiman, 1996).

The basic implementation of TMLE in the R-package **tmle** uses by default three algorithms:
1. SL libraries SL.glm (main terms logistic regression of A and W),
2. SL.step (stepwise forward and backward model selection using AIC criterion, restricted to second order polynomials) and,

3. SL.glm.interaction (a glm variant that include second order polynomials and two by two interactions of the main terms included in the model).

The principal interest of calling the Super-Learner is to obtain the less-unbiased estimated for $\bar{Q}_n^0(A, W)$ and $g_0(A, W)$. It is achieved by obtaining the smallest expected loss function for Y or A (binary outcomes), respectively. For instance, the negative logarithmic loss function for Y is computed as the minimizer of the expected squared error loss:

$$\bar{Q}_0 = \arg\min_{\bar{Q}} E_0 L(O, \bar{Q}),$$

where $L(O, \bar{Q})$ is:

$$(Y - \bar{Q}(A, W))^2$$

Note: see appendix one for a short introduction to the Super-Learner and ensemble learning thecniques.

1. **Step One**: $\bar{Q}_n^0(A, W)$ prediction

Hide

```
#Q0
library(SuperLearner)
#Specify SuperLearner libraries
SL.library <- c("SL.glm","SL.step","SL.glm.interaction")
#Data frame with X with baseline covariates and exposure
X <- subset(ObsData, select=c(A, w1, w2, w3, w4))
n <- nrow(ObsData)
#Create data frames with A=1 and A=0
X1<-X0<-X
X1$A <-1
X0$A <-0
#Create new data by stacking
newdata <- rbind(X,X1,X0)
#Call superlearner
Qinit <- SuperLearner(Y=ObsData$Y, X=X, newX=newdata, SL.library=SL.library, family="binomial")
Qinit
```

```
Call:
SuperLearner(Y = ObsData$Y, X = X, newX = newdata
, family = "binomial", SL.library = SL.library)

                           Risk   Coef
SL.glm_All               0.1766 0.6002
SL.step_All              0.1766 0.0000
SL.glm.interaction_All 0.1767 0.3998
```

Hide

```
#Predictions
#Pred prob of survival given A, W
QbarAW <- Qinit$SL.predict[1:n]
#Pred prob of surv for each subject given A=1 and
 w
Qbar1W <- Qinit$SL.predict[(n+1):(2*n)]
#Pred prob of surv for each subject given A=0 and
 w
Qbar0W <- Qinit$SL.predict[(2*n+1):(3*n)]
#Simple substitution estimator Psi(Q0)
PsiHat.SS <- mean(Qbar1W-Qbar0W);PsiHat.SS
```

```
[1] 0.199
```

2. **Step two**: $g_0(A, W)$ prediction

Hide

```
#Step 2 g_0(A|W) with SuperLearner
w <- subset(ObsData, select=c(w1,w2,w3,w4))
gHatSL <- SuperLearner(Y=ObsData$A, X=w, SL.libra
ry=SL.library, family = binomial)
gHatSL;mean(gHatSL)
```

```
Call:  SuperLearner(Y = ObsData$A, X = w, family
= binomial, SL.library = SL.library)

                           Risk   Coef
SL.glm_All               0.2091 0.0000
SL.step_All              0.2091 0.3803
SL.glm.interaction_All 0.2090 0.6197
```

```
argument is not numeric or logical: returning NA
```

```
[1] NA
```

Hide

```
#Generate the pred prob of A=1 and, A=0 given covariates
gHat1W <- gHatSL$SL.predict
gHat0W <- 1-gHat1W
#Step 3: Clever covariate
HAW <- as.numeric(ObsData$A==1)/gHat1W - as.numeric(ObsData$A==0)/gHat0W;mean(HAW)
```

```
[1] 0.002954
```

Hide

```
H1W <-  1/gHat1W
H0W <- -1/gHat0W
```

3. **Steps 3 and 4**: fluctuation step and substitution estimation for for $\bar{Q}_n^0(A, W)$ to $\bar{Q}_n^1(A, W)$

Hide

```
#Step 4: Substitution estimaiton Q* of the ATE.
logitUpdate <- glm(ObsData$Y ~ -1 + offset(qlogis(QbarAW))+HAW, family='binomial')
eps <- logitUpdate$coef;eps
```

```
      HAW 
0.0010483 
```

Hide

```
#Calculating the predicted values for each subject under each txt
QbarAW.star <- plogis(qlogis(QbarAW)+eps*HAW)
Qbar1W.star <- plogis(qlogis(Qbar1W)+eps*H1W)
Qbar0W.star <- plogis(qlogis(Qbar0W)+eps*H0W)
PsiHat.TMLE.SL <- mean(Qbar1W.star) - mean(Qbar0W.star)
cat("PsiHat.TMLE.SL:", PsiHat.TMLE.SL)
```

```
PsiHat.TMLE.SL: 0.1995
```

Hide

```
cat("\n PsiHat.TMLE.SL_bias:", abs(True_Psi-PsiHat.TMLE.SL))
```

```
PsiHat.TMLE.SL_bias: 0.001456
```

Hide

```
cat("\n Relative_PsiHat.TMLE.SL_bias:",abs(True_Psi-PsiHat.TMLE.SL)/True_Psi*100,"%")
```

```
Relative_PsiHat.TMLE.SL_bias: 0.7354 %
```

11 R-TMLE

Using the R-package **tmle**

Hide

```
library(tmle)
w <- subset(ObsData, select=c(w1,w2,w3,w4))
tmle <- tmle(Y, A, W=w)
cat("TMLER_Psi:", tmle$estimates[[2]][[1]],";","95%CI(", tmle$estimates[[2]][[3]],")")
```

```
TMLER_Psi: 0.1994 ; 95%CI( 0.1799 0.2189 )
```

Hide

```
cat("\n TMLE_bias:", abs(True_Psi-tmle$estimates[[2]][[1]]))
```

```
TMLE_bias: 0.001407
```

Hide

```
cat("\n Relative_TMLE_bias:",abs(True_Psi-tmle$estimates[[2]][[1]])/True_Psi*100,"%")
```

```
Relative_TMLE_bias: 0.7108 %
```

12 R-TMLE improving prediction

In addition to the default algorithms implemented in the R-tmle package, we can improve our estimation calling more efficient machine learing algorithms, such as generalized additive models and the Random Forest in this particular example:

Hide

```
SL.TMLER.Psi <- tmle(Y=Y, A=A, W=w, family="binomial",
     Q.SL.library = c("SL.glm", "SL.step", "SL.glm.interaction", "SL.gam", "SL.randomForest"),
     g.SL.library = c("SL.glm", "SL.step", "SL.glm.interaction", "SL.gam", "SL.randomForest"))
cat("SL.TMLER.Psi:", SL.TMLER.Psi$estimates[[2]][[1]],";","95%CI(", SL.TMLER.Psi$estimates[[2]][[3]],")")
```

```
SL.TMLER.Psi: 0.1994 ; 95%CI( 0.1799 0.2189 )
```

Hide

```
cat("\n SL.TMLER.Psi_bias:", abs(True_Psi-SL.TMLER.Psi$estimates[[2]][[1]]))
```

```
 SL.TMLER.Psi_bias: 0.001398
```

Hide

```
cat("\n Relative_SL.TMLER.Psi_bias:",abs(True_Psi-SL.TMLER.Psi$estimates[[2]][[1]])/True_Psi*100,"%")
```

```
 Relative_SL.TMLER.Psi_bias: 0.7061 %
```

We have demonstrated:

1. **TMLE excels** the AIPTW estimator and,
2. TMLE **best performance** is obtained when calling more advanvanced **Super-Learner** algorithms.

13 Appendix One

Efron in 1982 showed that the empirical **Influence Curve** estimate of standar error is the same as the one obtained using the infinitesimal jackknife and the nonparametric deltha method (Efron and Efron, 1982). 1. The Delat Method = First order of the Taylor series expansion:

$$f(x) \approx f(x) - f(\mu) + (x - \mu)f'(\mu);$$

where $f'(\mu)$ is the derivative of the function with respect to X evaluated at the mean of X. Therefore, squaring both terms, the variance is approximately:

$$\mathrm{E}[f(x) - f(\mu)]^2 \approx \mathrm{E}(x - \mu)^2 \times [f'(\mu)]^2;$$

The left-hand sie of the above equation is approximately the variance of $f(x)$ and applied to the empirical distribution of X, the sample estimate of variance for X replaces:

$$\mathrm{E}(x_i - \mu)^2.$$

The infinitesimal jackknife estimate of the standard error is defined as follows:

$$SD_{ij}(\theta_e) = \left(\frac{\sum_{i=1}^{n} U_i^2}{n^2}\right)^{1/2};$$

where θ_e is the estimate of the parameter θ and U_i is a **directional derivative** in the direction of the ith coordinate centered at the mean of the empirical distribution function.

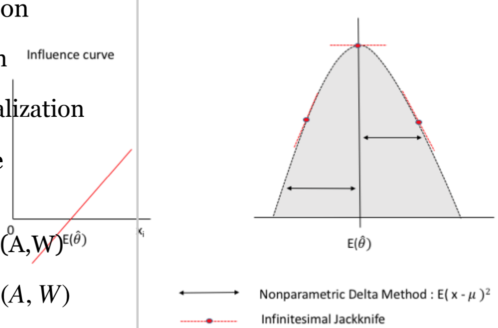

Figure 3: Estimate of the ψ Standard Error using the efficient Influence Curve.
Image credit: Miguel Angel Luque-Fernandez.

14 Appendix Two

With TMLE we can call the R-package **Super-Learner (SL)**. The *SL* uses **cross-validation** and **ensembled learning** (using all the predictions of multiple stacked learning algorithms) techniques to improve model prediction performance (Breiman, 1996).

The **SL** algorithm provides a system based on V-fold cross-validation (Efron and Gong, 1983) (10-folds) to combine adapttively multiple algorithms into an improved estimator, and returns a function we can also use for prediction in new datasets.

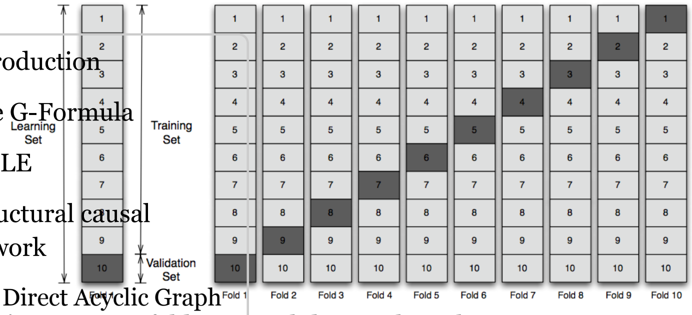

Figure 4: 10-fold cross-validation algorithm.
Source: Sherri Rose in Esembles in Public Health and Healt Policy. May 4, 2016.

The basic implementation of TMLE in the R-package **tmle** uses by default three algorithsm: 1. SL libraries SL.glm (main terms logistic regression of A and W),
2. SL.step (stepwise forward and backward model selection using AIC criterion, restricted to second order polynomials) and, 3. SL.glm.interaction (a glm variant that include second order polynomials and two by two interactions of the main terms included in the model).

The principal interest of calling the Super-Learner is to obtain the less-unbiased estimated for $\bar{Q}_n^0(A, W)$ and $g_0(A, W)$. It is achieved by obtaining the smallest expected loss function for Y or A (binary outcomes), respectively. For instance, the negative logarithmic loss function for Y is computed as the minimizer of the expected squared error loss:

$$\bar{Q}_0 = \text{arg min}_{\bar{Q}} E_0 L(O, \bar{Q}),$$

where $L(O, \bar{Q})$ is:

$$(Y - \bar{Q}(A, W))^2$$

The **SL** algorithm first split the data into 10 blocks and fits each of the selected algoriths on the training set (non-shaded blocks), then predicts the estimated probabilities of the outcome (Y) using the validation set (shaded block) for each algorithm, based on the corresponding training set. Afterwards, the **SL** estimates the the cross-validating riks for each algorithm averaging the risks across validation sets resulting in one estimated cross-validated risk for each algorithm. Finally, the **SL** select the combination of Z that minimizes the cross-validation risk, defined as the minimum mean square error for each of the selected algorithms using Y and Z. A weighted combination of the algorithms (ensemble learning) in Z is then used to predict the outcome (Y) (see Figure 5).

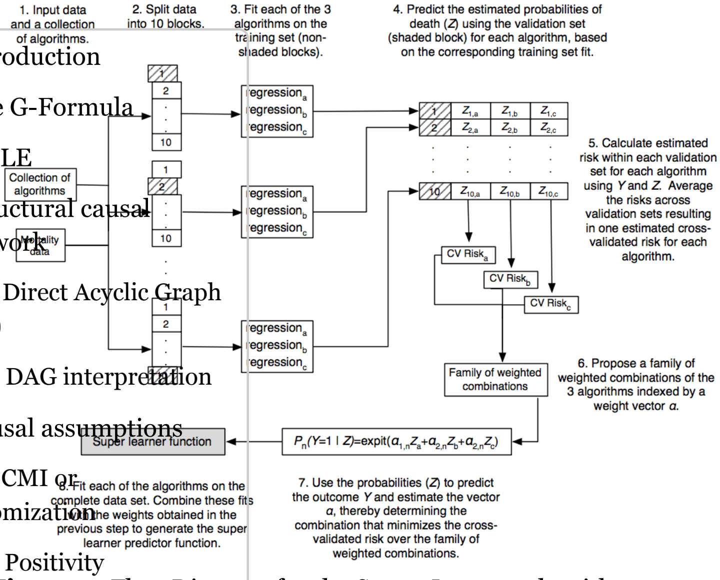

Figure 5: Flow Diagram for the Super-Learner algorithm.
Source: Mark van der Laan and Sherri Rose. Targeted learning: causal inference for observational and experimental dataSpringer Series in Statistics, 2011.

15 Session Info

Hide

```
devtools::session_info()
```

```
Session info -------------------------------------------------------------------
```

```
 setting  value
 version  R version 3.3.0 (2016-05-03)
 system   x86_64, darwin13.4.0
 ui       RStudio (1.0.44)
 language (EN)
 collate  en_US.UTF-8
 tz       Europe/London
 date     2016-11-01
```

```
Packages -----------------------------------------------------------------------
```

```
 package      * version   date       source
 assertthat     0.1       2013-12-06 CRAN (R 3.3.0
)
 base64enc      0.1-3     2015-07-28 CRAN (R 3.3.0
)
 codetools      0.2-14    2015-07-15 CRAN (R 3.3.0
)
 devtools       1.12.0    2016-06-24 CRAN (R 3.3.0
)
 digest         0.6.10    2016-08-02 CRAN (R 3.3.0
)
 DT           * 0.2       2016-08-09 CRAN (R 3.3.0
)
 evaluate       0.10      2016-10-11 CRAN (R 3.3.0
)
 foreach      * 1.4.3     2015-10-13 CRAN (R 3.3.0
)
 formatR        1.4       2016-05-09 CRAN (R 3.3.0
)
 gam          * 1.14      2016-09-10 CRAN (R 3.3.0
)
 htmltools      0.3.5     2016-03-21 CRAN (R 3.3.0
)
 htmlwidgets    0.7       2016-08-02 CRAN (R 3.3.0
)
 iterators      1.0.8     2015-10-13 CRAN (R 3.3.0
)
 jsonlite       1.1       2016-09-14 CRAN (R 3.3.0
)
 knitr          1.14      2016-08-13 CRAN (R 3.3.0
)
 magrittr       1.5       2014-11-22 CRAN (R 3.3.0
)
 memoise        1.0.0     2016-01-29 CRAN (R 3.3.0
)
 nnls         * 1.4       2012-03-19 CRAN (R 3.3.0
)
 randomForest * 4.6-12    2015-10-07 CRAN (R 3.3.0
)
 Rcpp           0.12.7    2016-09-05 CRAN (R 3.3.0
)
 rmarkdown      1.1.9007  2016-10-25 Github (rstud
io/rmarkdown@746d0eb)
 rprojroot      1.0-2     2016-03-28 CRAN (R 3.3.0
)
 rsconnect      0.5       2016-10-17 CRAN (R 3.3.0
)
 rstudioapi     0.6       2016-06-27 CRAN (R 3.3.0
)
 stringi        1.1.2     2016-10-01 CRAN (R 3.3.0
```

```
)
 stringr        1.1.0    2016-08-19 CRAN (R 3.3.0
)
 SuperLearner * 2.0-19   2016-02-04 CRAN (R 3.3.0
)
 tibble         1.2      2016-08-26 CRAN (R 3.3.0
)
 tmle         * 1.2.0-4  2014-03-09 CRAN (R 3.3.0
)
 withr          1.0.2    2016-06-20 CRAN (R 3.3.0
)
 yaml           2.1.13   2014-06-12 CRAN (R 3.3.0
)
```

16 Thank you

Thank you for participating in this tutorial.

If you have updates or changes that you would like to make, please send me (https://github.com/migariane/MALF) a pull request. Alternatively, if you have any questions, please e-mail me. You can cite this repository as:

Luque-Fernandez MA, (2016). Targeted Maximum Likelihood Estimation a Step by Step Guided Implementation. GitHub repository, http://migariane.github.io/TMLE.nb.html (http://migariane.github.io/TMLE.nb.html).

Miguel Angel Luque Fernandez
E-mail: *miguel-angel.luque at lshtm.ac.uk*
Twitter `@WATZILEI`

17 References

Breiman L. (1996). Stacked regressions. *Machine learning* **24**: 49–64.

Bühlmann P, Drineas P, Laan M van der, Kane M. (2016). Handbook of big data. CRC Press.

Efron B, Efron B. (1982). The jackknife, the bootstrap and other resampling plans. SIAM.

Efron B, Gong G. (1983). A leisurely look at the bootstrap, the jackknife, and cross-validation. *The American Statistician* **37**: 36–48.

Greenland S, Robins JM. (1986). Identifiability, exchangeability, and epidemiological confounding. *International journal of epidemiology* **15**: 413–419.

Gruber S, Laan M van der. (2011). Tmle: An r package for targeted maximum likelihood estimation. *UC Berkeley Division of Biostatistics Working Paper Series*.

Hampel FR. (1974). The influence curve and its role in robust estimation. *Journal of the American Statistical Association* **69**: 383–393.

Laan M van der, Rose S. (2011). Targeted learning: Causal inference for observational and experimental data. Springer Series in Statistics.

Neugebauer R, Laan M van der. (2005). Why prefer double robust estimators in causal inference? *Journal of Statistical Planning and Inference* **129**: 405–426.

Robins JM, Hernán MA, Brumback B. (2000). Marginal structural models and causal inference in epidemiology. *Epidemiology* 550–560.

Rubin DB. (2011). Causal inference using potential outcomes. *Journal of the American Statistical Association*.

Rubin DB. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology* **66**: 688.

Van der Laan MJ, Polley EC, Hubbard AE. (2007). Super learner. *Statistical applications in genetics and molecular biology* **6**.