

Stacked Generalization: An Introduction to Super Learning

Ashley I. Naimi, PhD^{1,*}

Laura B. Balzer, PhD²

¹ Department of Epidemiology, University of Pittsburgh.

² Department of Biostatistics and Epidemiology, University of Massachusetts, Amherst.

*Correspondence: Department of Epidemiology
University of Pittsburgh
130 DeSoto Street
503 Parran Hall
Pittsburgh, PA 15261
ashley.naimi@pitt.edu

Conflicts of Interest: None

Acknowledgements: We thank Susan Gruber and Mark J van der Laan for expert advice.

Sources of Funding: NIH grant number UL1TR001857 and R37AI051164

Text word count: 2,320 / 3,000

Abstract word count: 117 / 250

Number of Figures: 2

Number of Tables: 0

Number of References: 35

Running head: Super Learning

Abstract

Stacked generalization is an ensemble method that allows researchers to combine several different prediction algorithms into one. Since its introduction in the early 1990s, the method has evolved several times into what is now known as “Super Learner”. Super Learner uses V -fold cross-validation to build the optimal weighted combination of predictions from a library of candidate algorithms. Optimality is defined by a user-specified objective function, such as minimizing mean squared error or maximizing the area under the receiver operating characteristic curve. Although relatively simple in nature, use of the Super Learner by epidemiologists has been hampered by limitations in understanding conceptual and technical details. We work step-by-step through two examples to illustrate concepts and address common concerns.

KEY WORDS: Ensemble Learning; Machine Learning; Super Learner; Stacked Generalization; Stacked Regression.

KEY MESSAGES:

- Stacked generalization is a meta-learning algorithm that allows researchers to combine several different methods to improve predictive performance.
- Since its introduction in the early 1990s, stacked generalization has evolved both practically and theoretically into the Super Learner.
- The Super Learner combines predictions from several methods by using V -fold cross-validation and minimizing a user-specified loss function, such as prediction error, negative-log-likelihood, or rank loss.
- Under reasonable constraints, Super Learner is guaranteed to perform asymptotically as well as the best performing algorithm included in the candidate set of algorithms.
- In practice, Super Learner is a powerful tool to estimate complex, real-world relationships while avoiding unsubstantiated parametric assumptions and over-fitting.

Predicting health-related outcomes is a topic of major interest in clinical and public health settings. Despite numerous advances in methodology in the past two decades, clinical and population health research scientists continue to rely heavily on parametric (e.g., logistic) regression models for prediction. Often, only a single model is specified to generate predictions.

In the early 1990s, Wolpert developed an approach to combine several “lower-level” machine learning methods into a “higher-level” model with the goal of increasing predictive accuracy.¹ He termed the approach “stacked generalization”, which later became known as “stacking”. Later, Breiman demonstrated how stacking can be used to improve the predictive accuracy in a regression context, and showed that imposing certain constraints on the higher-level model improved predictive performance.² More recently, van der Laan and colleagues proved that stacking possesses certain ideal theoretical properties.^{3–5} In particular, their oracle inequality guarantees that in large samples the algorithm will perform at least as well as the best individual predictor included in the ensemble. Therefore, choosing a large library of diverse algorithms will enhance performance, and creating the best weighted combination of candidate algorithms will further improve performance. Here, “best” is defined in terms of a bounded loss function, and over-fitting is avoided with V -fold cross-validation. In this context, the term “Super Learner” was coined.

Super Learner has tremendous potential for improving the quality of prediction algorithms in applied health sciences, and minimizing the extent to which empirical findings rely on parametric modeling assumptions. While the number of simulation studies and applications of Super Learner are growing,^{6–10} wider use may be hampered by comparatively few pedagogic examples. Here, we add to prior work by providing step-by-step implementation in the context of two simple simulations. Our examples are motivated by common challenges to estimate a dose-response curve, and to build a classifier for a binary outcome. We also link Super Learner to prior work on stacking, provide guidelines on fitting the Super Learner to empirical data, and discuss common concerns. Full R code¹¹ is publicly available at [GitHub](#).

Example 1: Dose-Response Curve

For 1,000 observations, we generate a continuous exposure X by drawing from a uniform distribution with a minimum of zero and a maximum of eight and then generate a continuous outcome

Y as

$$Y = 5 + 4 \times \sqrt{9X} \times \mathbb{I}(X < 2) + \mathbb{I}(X \geq 2) \times (|x - 6|^2) + \epsilon, \quad (1)$$

where $\mathbb{I}()$ denotes the indicator function evaluating to 1 if the argument is true (zero otherwise), and ϵ was drawn from a doubly-exponential distribution with mean zero and scale parameter one. The true dose-response curve is depicted by the black line in Figure 1. We now manually demonstrate how Super Learner can be used to flexibly model this relation without making parametric assumptions.

To simplify our illustration, we consider only two “level-zero” algorithms as candidates to the Super Learner library: generalized additive models with 5-knot natural cubic splines (`gam`)¹² and multivariate adaptive regression splines implemented via the `earth` package.¹⁷ In practice, a large and diverse library of candidate estimators is recommended. Our specific interest is in quantifying the mean of Y as a (flexible) function of X . To measure the performance of candidate algorithms and to construct the weighted combination of algorithms, we select the L_2 squared error loss function $(Y - \hat{Y})^2$ where \hat{Y} denotes our predictions. Minimizing the expectation of the L_2 loss function is equivalent to minimizing mean squared error, which is the same objective function used in ordinary least squares regression.^{13(section 2.4)} To estimate this expected loss, called the “risk”, we use V -fold cross-validation with $V = 5$ folds.

Step 1. Split the observed “level-zero” data into 5 mutually exclusive and exhaustive groups of $n/V = 1000/5 = 200$ observations. These groups are called “folds”.

Step 2. For each fold $v = \{1, \dots, 5\}$,

- a. Define the observations in fold v as the validation set, and all remaining observations (80% of the data) as the training set.
- b. Fit each algorithm on the training set.
- c. For each algorithm, use its estimated fit to predict the outcome for each observation in the validation set. Recall the observations in the validation set are not used to train each candidate algorithm.

- d. For each algorithm, estimate the risk. For the L_2 loss, we average the squared difference between the outcome Y and its prediction \hat{Y} for all observations in the validation set
 - v. In other words, we calculate the mean squared error (MSE) between the observed outcomes in the validation set and the predicted outcomes based on the algorithms fit on the training set.

Step 3. Average the estimated risks across the folds to obtain one measure of performance for each algorithm. In our simple example, the cross-validated estimates of the squared prediction error are 2.58 for `gam` and 2.48 for `earth`.

At this point, we could simply select the algorithm with smallest cross-validated risk estimate (here, `earth`). This approach is sometimes called the Discrete Super Learner.⁶ Instead, we combine the cross-validated predictions, which are referred to as the “level-one” data, to improve performance and build the “level-one” learner.

Step 4. Let $\hat{Y}_{\text{gam-cv}}$ and $\hat{Y}_{\text{earth-cv}}$ denote the cross-validated predicted outcomes from `gam` and `earth`, respectively. Recall the observed outcome is denoted Y . To calculate the contribution of each candidate algorithm to the final Super Learner prediction, we use non-negative least squares to regress the actual outcome against the predictions, while suppressing the intercept and constraining the coefficients to be non-negative and sum to 1:

$$\mathbb{E}(Y | \hat{Y}_{\text{gam-cv}}, \hat{Y}_{\text{earth-cv}}) = \alpha_1 \hat{Y}_{\text{gam-cv}} + \alpha_2 \hat{Y}_{\text{earth-cv}}, \quad (2)$$

such that $\alpha_1 \geq 0$; $\alpha_2 \geq 0$, and $\sum_{k=1}^2 \alpha_k = 1$. Combining the $\hat{Y}_{\text{gam-cv}}$ and $\hat{Y}_{\text{earth-cv}}$ under these constraints (non-negative estimates that sum to 1) is referred to as a “convex combination,” and is motivated by both theoretical results and improved stability in practice.^{2,5} Non-negative least squares corresponds to minimizing the mean squared error, which is our chosen loss function (and thus, fulfills our objective). We then normalize the coefficients from this regression to sum to 1. In our simple example, the normalized coefficient values are $\hat{\alpha}_1 = 0.387$ for `gam` and $\hat{\alpha}_2 = 0.613$ for `earth`. Therefore, both generalized additive models and regression splines each contribute approximately 40% and 60% of the weight in the optimal predictor.

Step 5. The final step is to use the above weights to generate the Super Learner, which can then be applied to new data (X) to predict the continuous outcome. To do so, re-fit `gam` and `earth` on the entire sample and denote the predicted outcomes as \hat{Y}_{gam} and \hat{Y}_{earth} , respectively. Then combine these predictions with the estimated weights from Step 4:

$$\hat{Y}_{\text{SL}} = 0.387 \hat{Y}_{\text{gam}} + 0.613 \hat{Y}_{\text{earth}} \quad (3)$$

where \hat{Y}_{SL} denotes our final Super Learner predicted outcome. The resulting predictions are shown in blue in Figure 1. For comparison, the predictions from the R package `SuperLearner-v2.0-23-9000` are shown in red,¹⁴ while the predictions from `gam` and `earth` are shown in light blue and green, respectively.

Example 2: Binary Classification

Our second example is predicting the occurrence of a binary outcome with goal of maximizing the area under the receiver operating characteristic (ROC) curve, which shows the balance between sensitivity and specificity for varying discrimination thresholds. For 10,000 observations, we generate five covariates $\mathbf{X} = \{X_1, \dots, X_5\}$ by drawing from a multivariate normal distribution and then generate the outcome Y by drawing from a Bernoulli distribution with probability

$$\begin{aligned} \mathbb{P}(Y = 1 | \mathbf{X}) = & 1 - \text{expit}\{\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5 \\ & + \boldsymbol{\beta}_1(X_1 : X_5)_1 + \boldsymbol{\beta}_2(X_1 : X_5)_2\}, \end{aligned}$$

where $\text{expit}(\bullet) = (1 + \exp[-\bullet])^{-1}$; and $(X_1 : X_5)_1$ and $(X_1 : X_5)_2$ denote all first and second order interactions between X_1, \dots, X_5 , respectively; and $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ denote a set of parameters, one for each interaction. In total, there were 25 terms plus the intercept in this model. The intercept was set to $\beta_0 = 2$, while all other parameters were drawn from a uniform distribution bounded by 0 and 1.

Our library of candidate algorithms consists of Bayesian GLMs (`bayesglm`), implemented via the `arm` package (v 1.9-3),¹⁵ and multivariate polynomial adaptive regression splines (`polymars`) via the `polyspline` package.¹⁶ Our objective is to generate an algorithm that correctly classifies individuals given covariates. Correct classification is a function of both sensitivity and specificity. Thus, to

measure the performance of these level-zero algorithms and build the meta-learner, we use the rank loss function, which corresponds to maximizing the area under the ROC curve (AUC).¹⁸ We again use 5-fold cross-validation to obtain an honest measure of performance and avoid over-fitting.

Steps 1-3. The implementation of Steps 1-3 are analogous to the previous example. In place of `gam` and `earth`, we use `bayesglm`, and `polymars`. In place of the L_2 loss function, we use the rank loss. Specifically, for each validation set and each algorithm, we estimate the sensitivity, specificity, and then compute the AUC. The expected loss (i.e., “risk”) can then be computed as $1 - \text{AUC}$. Averaging the estimated risks across the folds yields one measure of performance for each algorithm. In our simple example, the cross-validated risk estimates are 0.122 for `bayesglm` and 0.114 for `polymars`.

As before, we could simply select the model with lowest cross-validated risk estimate (here, `polymars`). Instead in Steps 4-5, we combine the resulting cross-validated predictions to generate the level-one learner.

Step 4. Let $\hat{Y}_{\text{bglm-cv}}$ and $\hat{Y}_{\text{pm-cv}}$ denote the cross-validated predictions from `bayesglm` and `polymars`, respectively. To calculate the contribution of each candidate algorithm to the final Super Learner prediction, use the rank loss function to define “optimal” as the convex combination that maximizes the AUC. Then estimate the α parameters in the following constrained regression

$$\mathbb{P}(Y = 1 | \hat{Y}_{\text{bglm-cv}}, \hat{Y}_{\text{pm-cv}}) = \alpha_1 \hat{Y}_{\text{bglm-cv}} + \alpha_2 \hat{Y}_{\text{pm-cv}}, \text{ where } \alpha_1 \geq 0; \alpha_2 \geq 0; \text{ and } \sum_{k=1}^2 \alpha_k = 1 \quad (4)$$

such that $(1 - \text{AUC})$ is minimized when comparing Super Learner predicted probabilities to the observed outcomes. These parameters can be estimated with a tailored optimization function, such as `optim`. In our simple example, the coefficient values are $\hat{\alpha}_1 = 0.223$ for `bayesglm` and $\hat{\alpha}_2 = 0.776$ for `polymars`.

Step 5. As before, the final step is to use the above coefficients to generate the Super Learner. To do so, refit `bayesglm` and `polymars` on the entire sample and denote the predicted outcomes as

\hat{Y}_{bglm} and \hat{Y}_{pm} . Then combine these predictions with the estimated weights:

$$\mathbb{P}(Y = 1 | \hat{Y}_{\text{bglm}}, \hat{Y}_{\text{pm}}) = 0.223 \hat{Y}_{\text{bglm}} + 0.776 \hat{Y}_{\text{pm}} \quad (5)$$

These predictions can then be used to compute the ROC curve displayed in blue in Figure 2.

For comparison the predictions from the R package SuperLearner are shown in red.

Discussion

Stacked generalizations, notably the Super Learner, are fast becoming an important part of the epidemiologic toolkit. There are several challenges in understanding precisely what stacking is, and how it is implemented. These challenges include the use of complex machine learning algorithms as candidate algorithms, and the actual process by which the resulting predictions are combined into the meta-learner. Here, we sought to clarify the latter aspect, namely, implementation of the Super Learner algorithm.

Several considerations (including strengths and limitations) merit attention. First, any number of loss functions could be chosen determine the optimal combination of algorithm-specific predictions. The choice should be based on the objective of the analysis. The target parameter depends on the loss function choice, but various loss functions can identify the same target parameter as minimizer of its risk. Our examples demonstrated the use of the L_2 loss to minimize prediction error when estimating a dose-response curve, and the rank loss to maximize AUC when developing a binary classifier. Other loss functions could be entertained. For example, Zheng *et al.* recently aimed to simultaneously maximize sensitivity and minimize rate of false positive predictions with application to identify high-risk individuals for pre-exposure prophylaxis.¹⁰

Second, a wide array of candidate algorithms can be included in the Super Learner library. We recommend including standard parametric modeling approaches (e.g., generalized linear models, simple mean, simple median) and more complex data-adaptive methods (e.g., penalized regression, and tree- or kernel-based methods). Often, the performance of data-adaptive methods depends on how tuning parameters are specified. The Super Learner can also include variations of the same algorithm under different tuning parameter choices (e.g., extreme gradient boosting with varying

shrinkage rates). We refer the readers to Polley et al.¹⁴ for a practical demonstration.

Third, a critically important part of Super Learning is the use of V -fold cross-validation. However, the optimal choice of V is not always clear. At one end of the extreme, leave-one-out cross validation chooses $V = N$, but is subject to potentially high variance and low bias. On the other end, 2-fold cross validation is subject to potentially low variance and high bias. A general rule of thumb is to use $V = 10$.¹⁹ Though common, this number will not optimize performance in all settings.²⁰ In general, we recommend increasing the number of folds (V) as sample size n decreases.

While the Super Learner with a rich set of candidates represents a versatile tool, important limitations should be noted, particularly in the context of effect estimation. First, no algorithm (Super Learner or any other machine learning method) should be used to replace careful thinking of the underlying causal structure. The goal of Super Learner is to do the best (as specified through the loss function) possible job predicting the outcome (or exposure) given the inputted covariates. Super Learner does not distinguish between confounders, instrumental variables, mediators, and the exposure. Indeed, Super Learner is a “black box” algorithm; so the exact contribution of each covariate to prediction is unclear. These contributions can be revealed by estimating variable importance measures, which quantify the marginal association between each predictor and the outcome after adjusting for the others. Nevertheless, a large predictor contribution may be the result of direct causation, unmeasured confounding, collider stratification, reverse causation, or some other mechanism.

The goal of prediction is distinct from causal effect estimation, but prediction is often an intermediate step in estimating causal effects.³¹ Indeed, some researchers have been advocating for the use of data-adaptive methods, including the Super Learner, for effect estimation via singly-robust methods, depending on estimation of either the conditional mean outcome or the propensity score.^{21–27} While flexible algorithms can reduce the risk of bias due to regression model misspecification, a serious concern is that the use of data-adaptive algorithms in this context can result in invalid statistical inference (i.e. misleading confidence intervals). Specifically, there is no theory to support the resulting estimator is asymptotically linear (i.e., consistent and asymptotically normal).^{28,31} This concern can be alleviated through the use of doubly robust estimators^{29–33} or

higher-order influence function based estimators.^{34,35} In contrast, statistical inference is usually not included in prediction algorithms¹³ (as in our examples). One could, however, evaluate the performance of Super Learner through an additional layer of cross-validation.⁶

Overall, Super Learner is an important tool that researchers can use to improve predictive accuracy, avoid overfitting, and minimize parametric assumptions. We have provided a simple explanation of the Super Learner to facilitate a more widespread use in epidemiology. More advanced treatments with realistic data examples are available^{5,7,31} and should be consulted for additional depth.

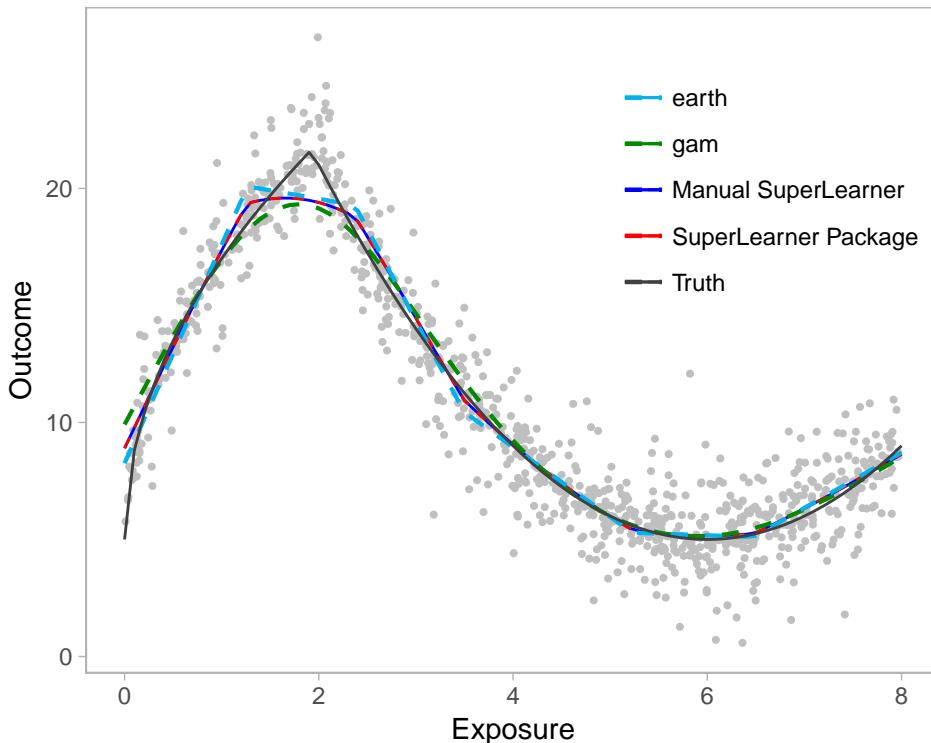


Figure 1: Dose-response curves for the relation between our simulated continuous exposure and continuous outcome in Example 1. The black line represents the true curve, while the red and blue lines represent curves estimated with the programmed Super Learner package in R, and the manually coded Super Learner. Light blue and green curves show the fits from the level-zero algorithms, earth and gam respectively. Gray dots represent observed data-points.

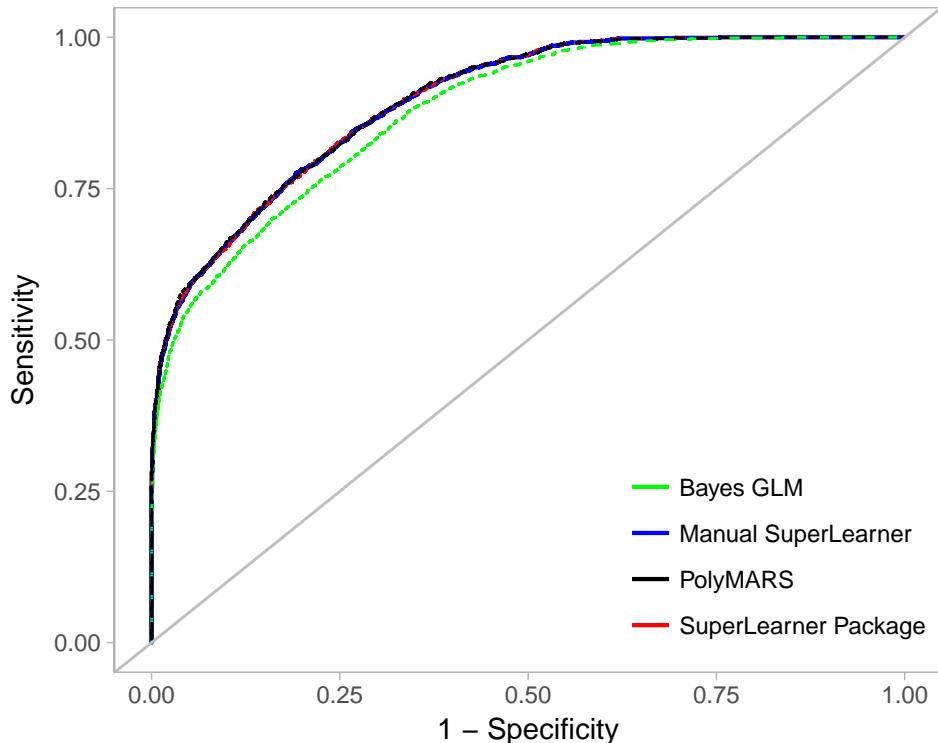


Figure 2: Receiver operating characteristic curves displaying the ability of 5 simulated exposures to predict the simulated outcome in Example 2. Blue line represents the curve obtained from the Super Learner package. Red dotted line represents curve obtained from manually coded Super Learner. The green line represents the curve from level-zero Bayes GLM algorithm, and the black line represents the curve from PolyMARS.

References

1. Wolpert D. Stacked generalization. *Neural Networks*. 1992; **5**:241–59.
2. Breiman L. Stacked regressions. *Machine Learning*. 1996; **24**:49–64.
3. van der Laan M, and Dudoit S. Unified Cross-Validation Methodology For Selection Among Estimators and a General Cross-Validated Adaptive Epsilon-Net Estimator: Finite Sample Oracle Inequalities and Example. *Technical Report 30, Division of Biostatistics, University of California, Berkeley*. 2003.
4. van der Laan M, Dudoit S, and van der Vaart AW. The cross-validated adaptive epsilon-net estimator. *Statistics & Decision*. 2006; **24**:373.
5. van der Laan MJ, Polley EC, and Hubbard AE. Super learner. *Statistical Applications in Genetics and Molecular Biology*. 2007; **6**:Article 25.
6. Polley EC, Rose S, and van der Laan MJ. Super Learning. In: van der Laan MJ and Rose S (Eds.) *Targeted Learning: Causal Inference for Observational and Experimental Data*. New York Dordrecht Heidelberg London: Springer. 2011; 43–66.
7. Rose S. Mortality risk score prediction in an elderly population using machine learning. *Am J Epidemiol*. 2013; **177**:443–452.
8. Pirracchio R, Petersen ML, Carone M, Resche Rigon M, Chevret S, and van der Laan M. Mortality prediction in intensive care units with the Super ICU Learner Algorithm (SICULA): a population-based study. *Lancet Resp Med*. 2015; **3**:42-52.
9. Petersen M, LeDell E, Schwab J, Sarovar MS, and *et. al.* Super Learner Analysis of Electronic Adherence Data Improves Viral Prediction and May Provide Strategies for Selective HIV RNA Monitoring. *J Acquir Immune Defic Syndr*. 2015; **69**:109-118.
10. Zheng W, Balzer L, van der Laan M, Petersen M, and the SEARCH Collaboration. Constrained binary classification using ensemble learning: an application to cost-efficient targeted PrEP strategies. *Statistics in Medicine*. 2017; **Early View**.

11. R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. 2017.
12. Hastie T and Tibshirani R. *Generalized Additive Models*. London; New York: Chapman & Hall. 1990.
13. Hastie T, Tibshirani R, and Friedman JH. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer. 2009.
14. Polley E, LeDell E, Kennedy C, and van der Laan M. *Super Learner: Super Learner Prediction*. 2016. URL <https://CRAN.R-project.org/package=SuperLearner>. R package version 2.0-22.
15. Andrew Gelman and Yu-Sung Su. *arm: Data Analysis Using Regression and Multi-level/Hierarchical Models*. 2016. URL <https://CRAN.R-project.org/package=arm>. R package version 1.9-3.
16. Charles Kooperberg. *polspline: Polynomial Spline Routines*. 2015. URL <https://CRAN.R-project.org/package=polspline>. R package version 1.1.12.
17. Stephen Milborrow. Derived from mda:mars by Trevor Hastie and Rob Tibshirani. Uses Alan Miller's Fortran utilities with Thomas Lumley's leaps wrapper.. *earth: Multivariate Adaptive Regression Splines*. 2017. URL <https://CRAN.R-project.org/package=earth>. R package version 4.5.0.
18. Erin Ledell. *Scalable Ensemble Learning and Computationally Efficient Variance Estimation*. 2015. PhD Dissertation. UC Berkeley.
19. Kohavi R. A study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*. 1995; 2.
20. Zhang Y and Yang Y. Cross-validation for selecting a model selection procedure. *Journal of Econometrics*. 2015; **187**.

21. Moodie EEM and Stephens DA. Treatment prediction, balance, and propensity score adjustment. *Epidemiology*. 2017; **28**.
22. Pirracchio R and Carone M. The balance super learner: A robust adaptation of the super learner to improve estimation of the average treatment effect in the treated based on propensity score matching. *Stat Methods Med Res*. 2016; :962280216682055.
23. McCaffrey DF, Griffin BA, Almirall D, Slaughter ME, Ramchand R, and Burgette LF. A tutorial on propensity score estimation for multiple treatments using generalized boosted models. *Stat Med*. 2013; **32**:3388–3414.
24. Westreich D, Lessler J, and Funk MJ. Propensity score estimation: neural networks, support vector machines, decision trees (cart), and meta-classifiers as alternatives to logistic regression. *J Clin Epidemiol*. 2010; **63**:826 – 833.
25. Lee BK, Lessler J, and Stuart EA. Improving propensity score weighting using machine learning. *Stat Med*. 2010; **29**:337–346.
26. Snowden JM, Rose S, and Mortimer KM. Implementation of g-computation on a simulated data set: Demonstration of a causal inference technique. *Am J Epidemiol*. 2011; **173**:731–738.
27. Westreich D, Edwards JK, Cole SR, Platt RW, Mumford SL, and Schisterman EF. Imputation approaches for potential outcomes in causal inference. *International Journal of Epidemiology*. 2015; :Published ahead of print July 25, 2015.
28. van der Vaart A. Higher Order Tangent Spaces and Influence Functions. *Statistical Science*. 2014; **29**:679–686.
29. Robins JM, Rotnitzky A, and Zhao LP. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American Statistical Association*. 1994; **89**:846-866.
30. van der Laan MJ, and Robins JM. *Unified Methods for Censored Longitudinal Data and Causality*. New York Berlin Heidelberg: Springer-Verlag. 2003.

31. van der Laan MJ and Rose S. *Targeted learning: causal inference for observational and experimental data*. New York, NY: Springer. 2011.
32. Naimi AI and Kennedy EH. Nonparametric double robustness. *bioRxiv*. 2017; .
33. Kennedy EH and Balakrishnan S. Discussion of “Data-driven confounder selection via Markov and Bayesian networks” by Jenny Häggström. *Biometrics*. 2017; **In Press**.
34. Robins J, Li L, Tchetgen Tchetgen E, and van der Vaart A. Higher order influence functions and minimax estimation of nonlinear functionals. In: Nolan D and Speed T (Eds.) *Probability and Statistics: Essays in Honor of David A. Freedman*, vol. Volume 2 of *Collections*. Beachwood, Ohio, USA: Institute of Mathematical Statistics. 2008; 335–421. URL <http://dx.doi.org/10.1214/193940307000000527>.
35. Diaz I, Carone M, and van der Laan MJ. Second-order inference for the mean of a variable missing at random. 2016; **12**:333. URL <http://www.degruyter.com/view/j/ibz.2016.12.issue-1/ibz-2015-0031/ibz-2015-0031.xml>.