PharmaSUG 2025 - Paper ST-171

Super Learner for Predictive Modeling and Causal Analysis

Honghe Zhao, Clay Thompson, and Michael Lamm, SAS Institute Inc.

ABSTRACT

Model misspecification is a common challenge in causal analysis with real-world observational data, where statistical models are used to understand the complex relationships among outcomes, exposures, and baseline characteristics. Misspecified models can easily lead to biased estimates of causal effects and incorrect conclusions. The super learner algorithm tackles this problem by enabling you to consider multiple candidate models with different specifications and combine them into a single, robust model. The super learner leverages the strengths of these candidate models to learn the complex relationships between the variables, reducing the risk of overfitting and model misspecification. This makes it well-suited for applications such as patient outcome predictions and causal analysis using real-world observational data, where complex interactions and dependencies among the variables are present. This paper discusses the implementation of the super learner algorithm in SAS® Visual Statistics software by using the SUPERLEARNER procedure. It also illustrates how you can use this procedure to build ensemble models for both predictive and causal analysis tasks.

INTRODUCTION

The super learner algorithm is an ensemble modeling technique that combines many different models to improve prediction accuracy. It uses cross-validation to create an optimal weighted combination of these models. Given the increased amount of observational and nonexperimental data and causal analysis techniques in the last few decades, the super learner has played an ever-growing role in predictive modeling and evaluating treatment strategies. The super learner algorithm has demonstrated history of successfully using observational data to answer causal questions. For example, the super learner algorithm and targeted maximum likelihood estimation are part of an ongoing effort to analyze electronic health records data to improve existing systems in medical product safety surveillance (US Food and Drug Administration 2023). In the 2016 Atlantic Causal Inference Conference data analysis challenge, many of the top-ranking competitors used estimators of the treatment effect that were based on the super learner (Dorie et al. 2019). The success of super learner in causal analysis can be largely attributed to its ability to leverage the strength of a diverse set of models specified by the user to effectively approximate the complex causal relationships that underlie the data generating process. By combining many candidate models into a single model, it also addresses the issue of model selection in an automated fashion, alleviating the concern for choosing the correct model for specific modeling tasks. Indeed, the usefulness of super learner is not limited to causal analysis, as it can be broadly applied to virtually any data analysis tasks that involve predictive modeling. For instance, it has been used to improve prediction of novel drug-target interactions in drug discovery (Dick et al. 2022).

This paper introduces super learner ensemble models in SAS Visual Statistics and demonstrates how you can use the new SUPERLEARNER procedure to train super learner models and compute counterfactual outcome predictions. The rest of the paper proceeds as follows. First, some background on the super learner algorithm, the potential outcomes framework, and the targeted maximum likelihood estimation method is discussed. Next, an overview of the main features of the SUPERLEARNER procedure is presented. An example then illustrates how to use the procedure to predict counterfactual outcomes for causal analysis.

BACKGROUND

SUPER LEARNING

The process of combining several individual base models (base learners) with a meta model (meta learner) is commonly referred to as stacking. The original idea of stacking appeared in Wolpert (1992) under the name “stacked generalization”. Later, Leo Breiman clarified and formalized the implementation of stacking, where he suggested using a weighted combination of the base learners with non-negative

constraints (Breiman 1996). Although stacking was effective at improve predictive accuracy in practice, it lacked a clear theoretical basis. The theoretical basis for stacking – and in fact, its theoretical optimality – was established by van der Laan and colleagues (Van der Laan, Polley, and Hubbard 2007), who also contributed the new name “super learner”. They proved that, when it comes to prediction accuracy, the super learner will typically perform at least as well as any of the individual contributing base learners.

To train a super learner model, you first specify and train each of the base learners on the training data set. You then perform k-fold cross-validation on each of the base learners and combine the cross-validated predictions into a new feature matrix. The new feature matrix and the original response values are then used to train the meta learner. The meta learner is typically specified as a weighted combination of the base learners, where the weights are constrained to be nonnegative and summing to one. As a result, a trained super learner model consists of trained base learners and a trained meta learner, whose weights correspond to the contribution of each base learner in the super learner ensemble. To generate predictions using a trained super learner model, you first generate predictions from each of the base learners and then compute a weighted average of these predictions.

There are a few recommendations when you specify the super learner in practice. For a comprehensive discussion on this topic, see Phillips et al. (2023). In general, the improvement in performance depends on the diversity of the base learners that you specify. The gain will be little if you specify a small number of base learners or base learners that yield highly correlated predictions. Therefore, it is recommended that you include as many base learners as computationally feasible with different functional forms and different values for the hyperparameters. There is generally no cost in performance by including weak learners because they will likely be assigned zero weights by the meta learner and not contribute to the final super learner ensemble. For example, you can include in the base learner library parametric models like ordinary least squares regression, semiparametric models such as generalized additive models, and data adaptive machine learning algorithms such as Bayesian additive regression trees, gradient boosting methods, and neural networks. You can also use variable selection and dimension reduction techniques when the data is high dimensional.

POTENTIAL OUTCOMES

Data analysts often use the potential (counterfactual) outcomes framework to estimate the causal effect of interest. This framework assumes that each subject in the study population has potential outcomes defined at each possible level of the treatment. For example, given a binary treatment that takes on values 0 and 1, this framework hypothesize that each subject is associated with a pair of potential outcomes, Y(0) and Y(1), where Y(0) represents the potential outcome that would be observed had this subject been given treatment 0, and Y(1) is the potential outcome had the subject been given treatment 1. Under this framework, it becomes easy to define the causal parameter that answers the causal question of interest.

This framework also makes it possible to articulate the necessary assumptions needed for valid causal analysis with real-world observational data. In observational data, it is often the case that the pretreatment variables X affect both the treatment variable T and the outcome variable Y, in which case those pretreatment variables are referred to as confounders. One usual assumption made that makes causal analysis possible with observational data is that all the confounders are measured and are available in the observational data. Under this assumption, statistical adjustment methods have been developed to control for the confounders and obtain unbiased estimates of the causal effects. These methods often involve modeling the outcome Y or the probability of treatment T, or both.

Traditional adjustment methods often rely on parametric models such as linear regression and logistic regression to model the outcome and the probability of treatment. While parametric models are easy to fit and make the behavior of the downstream estimates easy to derive, they are prone to model misspecification. This is because the relationships between the causal quantities in real-world observational data are often highly complex and nonlinear, and in most cases the parametric models are not flexible enough that they fail to adequately approximate these relationships. Model misspecification is a real challenge when inferring causation from observational data. At best, adjustment methods with misspecified models can lead to a loss of statistical efficiency. At worst, it can produce biased estimates of a treatment effect and invalid confidence intervals.

The super learner provides a solution to the problem of model misspecification, because you can create a base learner library that includes a variety of data adaptive models that learn the complex data patterns in different ways without imposing heavy restrictions on the variable relationships. As a result, the super learner is an important ingredient used by many state-of-the-art adjustment methods for estimating various causal effect measures in observational studies.

TMLE

Under causal assumptions, you can use the super learner model to predict the counterfactual outcomes and obtain a valid estimate of the causal effect. However, this approach alone does not provide a recipe to assess the uncertainty of that estimate (i.e. you cannot obtain confidence intervals or P-values). Targeted maximum likelihood estimation (TMLE) is a framework that allows you to leverage data adaptive models like the super learner and still make valid inference about the causal effect of interest (Van der Laan and Rubin, 2006). The TMLE is attractive due to its double robustness property: if you can accurately model either the outcome Y or the probability of treatment T (and not necessarily both), then the final TMLE estimate would be unbiased (under proper causal assumptions). Moreover, if you accurately model both the outcome and the probability of treatment, the final estimate will also have the smallest possible variance. Therefore, it is recommended to use the super learner to model these two quantities so that the TMLE estimate will have the best possible chance to have small bias and variability.

THE SUPERLEARNER PROCEDURE

The SUPERLEARNER procedure is available in the 2024.12 and later releases of SAS Visual Statistics and in the 2025.02 and later releases of SAS Viya Workbench. This procedure allows you to train super learner models for continuous or binary response variables. The procedure also allows you to train a cross-validated selector (discrete super learner), which chooses from the base learner library a single base learner that has the best performance in cross-validation instead of the default ensemble super learner, which creates a convex combination of the base learners.

The SUPERLEARNER procedure supports a wide selection of model types that can be used to form a diverse library of base learners. To specify a base learner, you can simply specify a desired model type in the BASELEARNER statement. For each model type you choose, there are also many options that you can specify in the BASELEARNER statement to further customize the base learner. For example, the REGSELECT model type supports a variety of model selection options from the lasso, elastic net, to stepwise selection. The available model types are listed in Table 1.

Base learner model type	Description
BART	Bayesian additive regression trees model
BNET	Bayesian network model
FACTMAC	Factorization machine model
FOREST	Forest model
GAMMOD	Generalized additive model based on low-rank regression splines
GAMSELECT	Generalized additive model with model selection
GPCLASS	Gaussian process classification model
GPREG	Gaussian process regression model
GRADBOOST	Gradient boosting model
LIGHTGRADBOOST	Light gradient boosting machine model
LOGSELECT	Logistic regression model with model selection
REGSELECT	Ordinary linear least squares model with model selection
SVMACHINE	Support vector machine model
TREESPLIT	Tree-based statistical model

Table 1: Base Learner Model Types

PROC SUPERLEARNER chooses the number of folds (K) used in k-fold cross-validation data-adaptively according to the rule given by Phillips et al. (2023). This rule automatically determines K based on the type of response variable as well as the training data sample size. In general, a larger K results in increased computation time but allows the super learner to assess each base learner's performance more accurately. By adopting this rule, PROC SUPERLEARNER prioritizes choosing a moderately large K, while keeping model training computationally feasible. In addition, PROC SUPERLEARNER also allows you to carry out stratified cross-validation by specifying a stratification variable in the CROSSVALIDATION statement. For instance, when the response variable is binary, it can be beneficial to ensure its prevalence in each training set of cross-validation approximately matches the overall prevalence.

You can save a trained super learner model and later use it to obtain predicted values for new observations. The saved super learner model can also be used to score new observations by using each of the trained base learner models. This can be useful if you want to compare the performance between the super learner and each of the base learners on a separate test data set. Moreover, you can use PROC SUPERLEARNER to compute predictive margins (Lane and Nelder 1982), which are obtained by fixing the values of one or more predictor variables and averaging the predicted values over the distribution of the predictor variables in the input data. You can also create an output data set that contains these predicted values, which correspond to predictions of the potential outcomes in causal analysis.

EXAMPLE

The following example demonstrates how you can use PROC SUPERLEARNER to train super learner models for the outcome and treatment variables and generate predicted values for the counterfactual outcomes and probability of treatment. You can then use PROC CAEFFECT to process the predicted values and compute the TMLE estimate of the average treatment effect.

The *SmokingWeight* data set is a subset of the data from the NHANES I Epidemiologic Follow-Up Study (NHEFS) in Hernán and Robins (2020) and is also used in the examples for the CAUSALTRT procedure in SAS/STAT software. In this example, the aim of the causal analysis is to estimate on average, what is the effect of smoking cessation on change in body weight over a 10-year period? The data contains the following variables:

- *Activity*: level of daily activity, with values 0, 1, and 2
- *Age*: age in 1971
- *BaseWeight*: weight in kilograms in 1971
- *Education*: level of education, with values 0, 1, 2, 3, and 4
- *Exercise*: amount of regular recreational exercise, with values 0, 1, and 2
- *PerDay*: number of cigarettes smoked per day in 1971
- *Quit*: 1 if an individual quit smoking between the initial and follow-up interviews; 0 otherwise
- *Race*: 0 for white; 1 otherwise
- *Sex*: 0 for male; 1 for female
- *Weight*: weight in kilograms at the follow-up interview
- *YearsSmoke*: number of years an individual has smoked
- *Change*: difference in weight at the follow-up and baseline interviews (measured in kilograms)

The following statements invoke the SUPERLEARNER procedure to fit a propensity score model for the treatment variable, *QUIT*, and create an output data set, *swTrtEstData*, which contains the predicted probabilities of treatment and copies of all variables in the *SmokingWeight* data set.

```
proc superlearner data=mylib.SmokingWeight seed=2324;
```

```
   target Quit / level=nominal;
   input Sex Race Education Exercise Activity / level=nominal;
   input Age YearsSmoke PerDay / level=interval;
   baselearner 'logistic' logselect;
   baselearner 'logistic_2way' logselect
      class=(Sex Race Education Exercise Activity)
      effect=(Age|YearsSmoke|PerDay|Sex|Race|Education|Exercise|Activity @ 2);
   baselearner 'logistic_ridge' logselect(selection=ELASTICNET(lambda=5 mixing=0))
      class=(Sex Race Education Exercise Activity)
      effect=(Age|YearsSmoke|PerDay|Sex|Race|Education|Exercise|Activity @ 2);
   baselearner 'gam' gammod;
   baselearner 'bart' bart(nTree=10 nMC=100);
   output out=mylib.swTrtEstData copyvars=(_ALL_);
run;
```

You specify the treatment variable as the response variable in the TARGET statement, and you specify the covariates in the INPUT statements. You use the LEVEL= option to specify whether the variables included in these statements should be treated as nominal or interval variables.

In this example, you specify five base learners using the BASELEARNER statement. The BASELEARNER statement requires that you specify a name for the base learner and a model type. For each model type, you can use options to further customize base learner specifications. Three of the base learners use the LOGSELECT model type with different model effects and model selection methods. The 'logistic' base learner fits a logistic regression model using covariates and response specified in the TARGET and INPUT statements. The 'logistic_2way' base learner also fits a logistic regression but includes two-way interactions among all the covariates. The 'logistic_ridge' learner adds a ridge penalty to the 'logistic_2way' learner. The 'gam' base learner specifies a generalized additive model, and the 'bart' base learner specifies a Bayesian additive regression trees model with 10 trees and 100 monte carlo iterations.

As mentioned earlier, the number of folds used for training is determined according to the rule given in Phillips et al. (2023). The method used to fit the meta learner model, the number of folds, and other basic fitting information of the super learner ensemble are displayed in Figure 1.

Model Information	
Data Source	SMOKINGWEIGHT
Target Variable	Quit
Target Variable Type	NOMINAL
Number of Folds	10
Number of Base Learners	5
Loss Function	Log Loss
Meta-learning Method	Convex-constrained Binomial Likelihood Maximization
Random Number Seed	2324

Figure 1: Treatment Assignment Model Information

The estimated weight of each base learner in the super learner ensemble for the treatment assignment model is displayed in Figure 2. Note that the weights are nonnegative and sum to 1 because the meta learning method implements a convex constraint. Base learner 'logistic_ridge' and 'bart' receive zero weights so they do not contribute to the super learner ensemble. In this case, the simplest logistic

regression model yields the largest super learner coefficient of 0.62.

Super Learner Model Coefficients

Name	Model Type	Coefficient	Cross-Validated Risk
logistic	Logistic Regression	0.62379	0.55602
logistic_2way	Logistic Regression	0.18984	0.59092
logistic_ridge	Logistic Regression	0	0.56960
gam	Generalized Additive Model	0.18638	0.56010
bart	Bayesian Additive Regression Trees	0	0.55771

Figure 2: Treatment Assignment Model Super Learner Coefficients

By default, PROC SUPERLEARNER models the probability of not quitting smoking, i.e. *Quit*=0. You specify the OUTPUT statement to save the predicted values into a data set named *swTrtEstData*. The COPYVARS option copies all the variables in the input data to the output data set. Alternatively, you can use the STORE statement to save the model as an item store and later invoke the SUPERLEARNER procedure to obtain predictions.

The name of the variable in the output data table that contains the predicted probability is *P_Quit0*. The following DATA step creates another variable *P_Quit1*, which contains the predicted probability of being assigned to the treatment condition, i.e. *Quit=1*:

```
data mylib.swTrtEstData;
  set mylib.swTrtEstData;
  P_Quit1 = 1 - P_Quit0;
run;
```

The following statements invoke PROC SUPERLEARNER again to fit a super learner model for the outcome variable *Change* as a function of all the baseline covariates and the treatment variable *Quit*:

```
proc superlearner data=mylib.SmokingWeight seed=2324;
   baselearner 'linear' regselect;
   baselearner 'linear_2way' regselect
      class=(Sex Race Education Exercise Activity Quit)
      effect=(Age|YearsSmoke|PerDay|Sex|Race|Education|Exercise|Activity|Quit @ 2);
   baselearner 'linear_ridge' regselect(selection=ELASTICNET(lambda=5 mixing=0))
      class=(Sex Race Education Exercise Activity Quit)
      effect=(Age|YearsSmoke|PerDay|Sex|Race|Education|Exercise|Activity|Quit @ 2);
   baselearner 'gam' gammod;
   baselearner 'bart' bart(nTree=10 nMC=100);
   margin 'quitsmoking' Quit=1;
   margin 'noquitsmoking' Quit=0;
   target Change / level=interval;
   input Sex Race Education Exercise Activity Quit / level=nominal;
   input Age YearsSmoke PerDay / level=interval;
   output out=mylib.swDREstData marginpred copyvars=(Quit Change P_Quit1 P_Quit0);
run;
```

Again, five base learners are specified, except the three base learners with the LOGSELECT model type now use the REGSELECT model type for modeling a continuous outcome variable. The treatment variable *QUIT* is also included as a covariate in the outcome model.

You use the OUTPUT statement again to create an output data set, *swDREstData*, which contains the predicted counterfactual outcomes. This is achieved by using the MARGIN statement. The MARGIN

statement requires that you specify a name for the predictive margin, the variables that define the predictive margin, and the values they are set to. The predictive margin named ‘quitsmoking’ sets the value of Quit to 1, which corresponds to the counterfactual scenario where subjects quit smoking; similarly, the other predictive margin named ‘noquitsmoking’ defines the counterfactual scenario where subjects continue to smoke. The predicted counterfactual outcomes are the predicted weight change if subjects quit smoking and if the subjects continue to smoke.

The “Model Information” table in Figure 3 summarizes basic information of the outcome model specification. Since the outcome variable is continuous, the convex-constrained least squares method is used for fitting the meta learner.

Model Information	
Data Source	SWTRTESTDATA
Target Variable	Change
Target Variable Type	INTERVAL
Number of Folds	10
Number of Base Learners	5
Loss Function	Quadratic Loss
Meta-learning Method	Convex-constrained Least Squares
Random Number Seed	2324

Figure 3: Outcome Model Information

Figure 4 below displays the estimated weight of each base learner in the super learner ensemble for the outcome model. While the first base learner received a zero weight, all other base learners contribute to the final super learner ensemble.

Super Learner Model Coefficients

Name	Model Type	Coefficient	Cross-Validated Risk
linear	Ordinary Linear Least Squares	0	57.31210
linear_2way	Ordinary Linear Least Squares	0.18160	59.30354
linear_ridge	Ordinary Linear Least Squares	0.15837	57.85823
gam	Generalized Additive Model	0.40464	56.70110
bart	Bayesian Additive Regression Trees	0.25539	56.93869

Figure 4: Outcome Model Super Learner Coefficients

The following statements use the CAEFFECT procedure to compute estimates of the potential outcome means and their difference, which corresponds to the average treatment effect of quitting smoking. The INFERENCE option requests standard errors and confidence intervals to be computed. The TMLE adjustment method is specified in the METHOD= option, but the CAEFFECT also supports other adjustment methods such as AIPW. To use the TMLE method, the TREATVAR and OUTCOMEVAR statements are required. Variables that contain predictions of the counterfactual outcomes (*quitsmoking*, *noquitsmoking*) and the treatment assignment probabilities (*P_Quit1*, *P_Quit0*) are specified in the POM statements. These predictions are combined by the doubly robust TMLE method for estimation. You request that the difference between the potential outcome mean estimates by specifying the DIFFERENCE statement.

```
proc caeffect data=mylib.swDREstData inference method=TMLE;
   treatvar Quit;
   outcomevar Change;
   pom treatLev=1 treatProb=P_Quit1 predout=quitsmoking;
   pom treatLev=0 treatProb=P_Quit0 predout=noquitsmoking;
   difference evtLev=1;
run;
```

The estimated potential outcome means their difference, along with the standard error, confidence intervals, test statistic and p-values, are displayed in Figure 5. If we were to assume that the covariates make up for a valid adjustment set for studying the effect of quitting smoking on weight change, then the difference in potential outcome mean estimates would correspond to an average treatment effect estimate of 3.22 kilograms increase in weight if the subjects quit smoking.

POM Estimates

Treatment Level	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
1	5.04444	0.32156	4.41420	5.67469	15.69	<.0001
0	1.82240	0.23385	1.36407	2.28073	7.79	<.0001

POM Differences

Treatment Levels		Difference	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Event	Reference						
1	0	3.2220	0.38595	2.46559	3.97850	8.35	<.0001

Figure 5: Estimates of Potential Outcome Means and Their Difference

CONCLUSION

The super learner algorithm considers models with different specifications and data adaptively computes the optimal combination, thereby minimizing model misspecification. This makes it well suited for approximating complex variable relationships that are ubiquitous in predictive modeling and causal analysis with real-world observational data. The SUPERLEARNER procedure in SAS Visual Statistics makes it easy to train a super learner model and use it to generate predictions. If your goal is to estimate

causal effects, the MARGIN statement allows you to obtain counterfactual outcome predictions so that you can combine them with different adjustment methods.

REFERENCES

US Food and Drug Administration. 2023. *Using artificial intelligence & machine learning in the development of drug & biological products: Discussion paper and request for feedback*.

Dorie, Vincent, et al. "Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition." *Statistical Science* 34.1 (2019): 43-68.

Dick, Kevin, et al. "Reciprocal perspective as a super learner improves drug-target interaction prediction (musdti)." *Scientific Reports* 12.1 (2022): 13237.

Wolpert, David H. "Stacked generalization." *Neural networks* 5.2 (1992): 241-259.

Breiman, Leo. "Stacked regressions." *Machine learning* 24 (1996): 49-64.

Van der Laan, Mark J., Eric C. Polley, and Alan E. Hubbard. "Super learner." *Statistical applications in genetics and molecular biology* 6.1 (2007).

Phillips, Rachael V., et al. "Practical considerations for specifying a super learner." *International Journal of Epidemiology* 52.4 (2023): 1276-1285.

Rubin, Donald B. "Estimating causal effects of treatments in randomized and nonrandomized studies." *Journal of educational Psychology* 66.5 (1974): 688.

Van der Laan, Mark J., and Daniel Rubin. "Targeted maximum likelihood learning." *The international journal of biostatistics* 2.1 (2006).

Hernán, M. A., and Robins, J. M. (2020). Causal Inference: What If. Boca Raton, FL: Chapman & Hall/CRC.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Honghe Zhao
SAS Institute Inc
Joe.Zhao@sas.com