# SuperLearner

**Eric C. Polley**
Biometric Research Branch
National Cancer Institute

**Mark J. van der Laan**
Division of Biostatistics
University of California, Berkeley

---

**Abstract**

An R package for the Super Learner van der Laan, Polley, and Hubbard (2007) is presented.

*Keywords*: prediction models, cross-validation, R.

---

## 1. Introduction

This vignette explains how to use the **SuperLearner** R package. The **SuperLearner** package provides a syntax and structure to implement the super learner algorithm van der Laan *et al.* (2007). A characteristic of the super learner algorithm is the ability to combine many different prediction algorithms together and let the data decide on the optimal ensemble. R is the perfect language for such an algorithm because of the wealth of available prediction algorithms already available in the community. One problem is that prediction algorithms do not have a common syntax, so one of the goals of the **SuperLearner** package is to translate these prediction algorithms into a common syntax to allow for easy programming of the super learner.

## 2. The Super Learner Algorithm

Table 1: Details of prediction algorithm wrappers

| Function | Package | Tuning Parameters | Description |
|----------|---------|-------------------|-------------|
| bagging | **ipred** | nbagg<br>minsplit<br>cp<br>maxdepth | Bagging CART trees |

(Continued on next page)

Table 1: Details of prediction algorithm wrappers

| Function | Package | Tuning Parameters | Description |
|---|---|---|---|
| bart | **BayesTree** | ntree | Bayesian Regression Trees |
| | | sigdf | |
| | | sigquant | |
| | | k | |
| | | power | |
| | | base | |
| | | ndpost | |
| | | nskip | |
| bayesglm | **arm** | prior.mean | Bayesian glm |
| | | prior.scale | |
| | | prior.df | |
| cforest | **party** | ntree | Conditional Tree Forest |
| | | mtry | |
| | | mincriterion | |
| | | teststat | |
| | | testtype | |
| | | replace | |
| | | fraction | |
| cv.spls | **spls** | K | Sparse partial least squares |
| | | eta | |
| DSA | **DSA** | maxsize | Deletion\Substitution\Addition |
| | | maxorderint | |
| | | maxsumofpow | |
| | | Dmove | |
| | | Smove | |
| | | vfold | |
| earth | **earth** | degree | Adaptive Regression Splines |
| | | penalty | |
| | | nk | |
| | | thresh | |
| | | minspan | |
| | | newvar.penalty | |
| | | fast.k | |
| | | fast.beta | |
| | | nfold | |
| | | pmethod | |
| gam | **gam** | deg.gam | Generalized additive model |

(Continued on next page)

Table 1: Details of prediction algorithm wrappers

| Function | Package | Tuning Parameters | Description |
|---|---|---|---|
| gbm | **gbm** | `gbm.trees`<br>`interaction.depth`<br>`cv.folds`<br>`shrinkage`<br>`n.minobsinnode`<br>`bag.fraction`<br>`train.fraction` | Gradient boosting |
| glm | **stats** | – | Generalized linear model |
| glmnet | **glmnet** | `alpha`<br>`lambda`<br>`nlambda`<br>`lambda.min`<br>`dfmax`<br>`type` | Elastic Net |
| knn | **class** | `k`<br>`use.all` | k-Nearest neighbors |
| loess | **stats** | `span`<br>`family`<br>`degree` | Local polynomial regression |
| logreg | **LogicReg** | `ntrees`<br>`nleaves`<br>`select`<br>`penalty`<br>`kfold`<br>`control` | Logic regression |
| mars | **mda** | `degree`<br>`nk`<br>`penalty`<br>`thresh`<br>`prune`<br>`forward.step` | Adaptive Regression Splines |
| nnet | **nnet** | `size`<br>`decay`<br>`rang` | Neural network |
| polymars | **polspline** | `maxsize`<br>`gcv`<br>`additive`<br>`knots`<br>`know.space` | Adaptive polynomial splines |
| polyclass | **polspline** | `maxdim`<br>`cv`<br>`additive` | Polychotomous regression |

(Continued on next page)

Table 1: Details of prediction algorithm wrappers

| Function | Package | Tuning Parameters | Description |
|---|---|---|---|
| randomForest | **randomForest** | ntree<br>mtry<br>nodesizes<br>sampsize<br>replace<br>maxnodes | Random Forest |
| Ridge | **MASS** | lambda | Ridge regression |
| rpart | **rpart** | cp<br>minsplit<br>xval<br>maxdepth<br>minbucket | Regression tree |
| step | **stats** | scope<br>scale<br>direction<br>steps<br>k | Stepwise regression |
| step.plr | **stepPlr** | type<br>lambda<br>cp<br>max.terms | Stepwise penalized logistic |
| svm | **e1071** | type<br>kernel<br>nu<br>degree<br>gamma<br>coef0<br>cost<br>cachesize<br>tolerance<br>epsilon<br>cross | Support vector machine |

# 3. Using the SuperLearner Package

```
> library(SuperLearner)
```

## 3.1. Creating prediction wrappers

A full list of the built-in prediction wrappers can be found with the function:

```
> listWrappers(what = 'SL')
```

```
 [1] "SL.DSA"             "SL.bart"
 [3] "SL.bayesglm"        "SL.caret"
 [5] "SL.caret.rpart"     "SL.cforest"
 [7] "SL.earth"           "SL.gam"
 [9] "SL.gbm"             "SL.glm"
[11] "SL.glm.interaction" "SL.glmnet"
[13] "SL.ipredbagg"       "SL.knn"
[15] "SL.loess"           "SL.logreg"
[17] "SL.mean"            "SL.nnet"
[19] "SL.polymars"        "SL.randomForest"
[21] "SL.ridge"           "SL.rpart"
[23] "SL.step"            "SL.step.forward"
[25] "SL.step.interaction" "SL.stepAIC"
[27] "SL.svm"             "SL.template"
```

And the included template creator:

```
> write.SL.template(file = '')
```

```
SL.template <- function(Y, X, newX, family, obsWeights, id, ...) {
  # load required packages
  # require('pkg')
  if(family$family == 'gaussian') {

  }
  if(family$family == 'binomial') {

  }
  # pred is the predicted responses for newX (on the scale of the outcome)
  pred <- numeric()
  # fit returns all objects needed for predict.SL.template
  fit <- list(object = )
  # declare class of fit for predict.SL.template
  class(fit) <- 'SL.template'
  # return a list with pred and fit
  out <- list(pred = pred, fit = fit)
  return(out)
}
```

### 3.2. Creating screening wrappers

A full list of the built-in screening wrappers can be found with the function:

```
> listWrappers(what = 'screen')
```

```
[1] "All"
[1] "screen.SIS"             "screen.corP"
[3] "screen.corRank"        "screen.glmnet"
[5] "screen.randomForest"   "screen.template"
[7] "screen.ttest"          "write.screen.template"
```

And the included template creator:

```
> write.screen.template(file = '')


screen.template <- function(Y, X, family, obsWeights, id, ...) {
  # load required packages
  # require('pkg')
  if (family$family == 'gaussian') {

  }
  if (family$family == 'binomial') {

  }
  # whichVariable is a logical vector,
  # TRUE indicates variable will be used
  whichVariable <- rep(TRUE, ncol(X))
  return(whichVariable)
}
```

### 3.3. Creating methods wrappers

The included template creator:

```
> write.method.template(file = '')


method.template <- function() {
  out <- list(
    # require allows you to pass a character vector with required packages
    # use NULL if no required packages
    require = NULL,

    # computeCoef is a function that returns a list with two elements:
    # 1) coef: the weights (coefficients) for each algorithm
    # 2) cvRisk: the V-fold CV risk for each algorithm
    computeCoef = function(Z, Y, libraryNames, obsWeights, control, verbose, ...) {
      cvRisk <- numeric()
      coef <- numeric()
      out <- list(cvRisk = cvRisk, coef = coef)
      return(out)
    },
```

```
  # computePred is a function that takes the weights and the predicted values
  # from each algorithm in the library and combines them based on the model to
  # output the super learner predicted values
  computePred = function(predY, coef, control, ...) {
    out <- crossprod(t(predY), coef)
    return(out)
  }
  )
  invisible(out)
}
```

# References

van der Laan MJ, Polley EC, Hubbard AE (2007). "Super Learner." *Statistical Applications in Genetics and Molecular Biology*, **6**(25).

# 4. Computing Environment

- R version 2.13.0 Patched (2011-04-21 r55579), `x86_64-apple-darwin9.8.0`

- Base packages: base, datasets, grDevices, graphics, methods, stats, utils

- Other packages: SuperLearner 2.0-4, nnls 1.3, quadprog 1.5-4

- Loaded via a namespace (and not attached): tools 2.13.0

**Affiliation:**

Eric Polley
Biometric Research Branch
National Cancer Institute
E-mail: eric.polley@nih.gov
URL: http://linus.nci.nih.gov/

Mark van der Laan
Division of Biostatistics
University of California, Berkeley
E-mail: laan@berkeley.edu
URL: http://www.stat.berkeley.edu/~laan/