Algorithm Description V\_2

SOCR CBDA Team

August 1, 2016

# SuperLearner Function

## SL.library: Algorithms description

### Notations:

* X: a predictors column-joined.
* Y: the response variable.
* k: the number of predictors.
* n: the number of observations. (see **???** for details)

### Generalized linear model (GLM)

GLM is a fitting generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution. This algorithm could be used on both categorical or discrete variables and continuous variables. In the meantime, instead of assuming that the responses should conform to normally distribution, the response variables could from any exponential families.

#### Advantages:

* This model could extend the linear model to problems in which the response is categorical or discrete rather than a continuous numeric variable. GLM does not assume a linear relationship between the dependent variable and the independent variables, but it does assume linear relationship between the transformed response in terms of the link function and the explanatory variables.
* In the Generalized Linear Model (GLM), the response variable may not normally distributed, . For instance, the response variable may be binary (logistic model). GLM addresses these cases – e.g., GLM logit and probit models are GLM special cases appropriate for dichotomous or polytomous variables.

#### Assumptions:

The data are independently distributed and are assumed from an exponential family.

#### Notations:

: The parameters of the linear predictor. GLM are typically fit to data by the method of **maximum likelihood**, using **iteratively weighted least squares procedure**.

#### Model components:

* A random component  
   The random component specifies the conditional distribution of the response variable, y, given the predictors. The conditional distributions are from an exponential family.
* A linear predictor   
   : is the value of the linear predictor. (x) represents the estimated value of the linear predictor.

Hence, the GLM models is shown as followed

* A link function
* The link function enables the connection of the predictor structural component to the response variable, since in some case may take on any value in , whereas the mean of a binary random variable must be in the interval (0,1). The link function translates from the scale of the mean response to the scale of the linear predictor.  
   is the mean of a conditional response distribution at a given point in the covariate space and is the result predicted from the linear predictor.
* The link function is important as it specifies the distribution of the response variable- e.g., "identify link" for normal and non-identity link function for non-normally distributed responses.  
  The choice of the link function is important and should be based on:

1. Knowledge of the response distribution,
2. Theoretical process assumptions, and
3. Empirical fit to the data.

* P.S. : In R, the glm function includes a family-generator function, which could match link functions to five standard exponential families automatically. Each family has its own canonical link, which is used by default if a link is not given explicitly.

|  |  |  |  |
| --- | --- | --- | --- |
| *Family* | *Default link* | *Range of y* | *Var(y│x)* |
| gaussian | identity |  |  |
| binomial | logit |  |  |
| poisson | log |  |  |
| Gamma | Inverse | (0,) |  |
| inverse.gaussian |  |  |  |

#### Statistical Analysis:

Assume the estimated value of the linear predictor as   
\* The estimated mean of the response is , because of the link function .  
\* The variance of distribution is , is the positive dispersion parameter related to the variance of the exponential family. For the binomial and poisson distributions, the dispersion parameter equal to 1, for Gaussian data, the parameters could be replaced by .

(see Fox, Weisberg, & Fox, 2011 for details)

### Generalized additive models (GAM)

GAM is a generalized linear model with a linear predictor, involving a sum of smooth functions of covariates. GAM in R uses **the backfitting algorithm** to combine different smoothing or fitting methods. **The Gauss-Seidel methods** are used to fit additive models by iteratively smoothing partial residuals. The smoothing or fitting methods currently supported are local regression and smoothing splines.

#### Advantages:

The model allows for rather flexible specification of the dependence of the response on the covariates by specifying the model only in terms of "smooth functions", rather than detailed parametric relationships.

#### Notation:

* : The smooth function for (feature i).

#### GAM in R:

In general the model has a structure like

where Y some exponential family distribution.

**A Gauss-Seidel method**:fitting additive models by iteratively smoothing partial residuals

The pseudo algorithm for this method is shown as follow.  
 Suppose the function is   
 1) Set and for .  
 2) Repeat steps 3 to 5 until the estimates, , stop changing.  
 3) For repeat steps 4 and 5.  
 4) Calculate partial residuals:  
 5)Set equal to the result of smoothing with respect to .

#### The smoothing or fitting methods:

##### local regression:

Combining much of the simplicity of linear least squares regression with the flexibility of nonlinear regression. It does this by fitting simple models to localized subsets of the data and building up a function that describes the deterministic part of the variation in the data, point by point.

Local regression model also be known as locally weighted polynomial regression. The polynomial is fitted using weighted least squares, giving more weight to points near the point whose response is being estimated and less weight to points further away.  
The traditional weight function is the tri-cube weight function

##### Smoothing splines:

The smoothing splines would circumvent the problem of knot selection, and simultaneously, they control for over-fitting by shrinking the coefficients of the estimated function.  
The estimators perform a regularized regression over the natural spline basis, placing knots at all points ,.  
Suppose the function is , where are the truncated power basis functions for natural splines with knots at .(the natural splines are only defined for odd orders )  
The coefficients are chosen , in order to minimize

where is the basis matrix defined as

, and is the penalty matrix defined as

Given the optimal coefficients minimizing , the smoothing spline estimate at x is defined as

* The parameter is a tuning parameter , often called the smoothing parameter. The higher the value of , the more shrinkage.
* Similar to least square regression, the is equal to
* Hence,

#### deg.gam

In our experiment, we choose different values of deg.gam. This notation represents the maximum degree of the polynomial function that going to be applied during the smoothing splines procedure.  
The default value of this is 2, but we reset this value equal to 1,2,3,4,5, respectively.

(see T. Hastie & Tibshirani, 1999 for details)  
(see Venables, Ripley, & Venables, 2002 for details)  
(see Wood, 2006 for details)

### A generalized linear model with lasso or elastic net regularization （GLMNET）

GLMNET is a model, which fit a generalized linear model via penalized maximum likelihood. The regularization path is computed for the **lasso** or **elastic net** penalty at a grid of values for the regularization parameter lambda. This model can deal with all shapes of data, including vary large sparse data matrices. Fits linear, logistic and multinomial, poisson, and Cox regression models.

#### Model component:

**The GLM part**: the GLM functions for different families are different.  
For “Gaussian” family is

For the other models are

**The penalty:**

when ,the penalty becomes the lasso penalty,  
when ,the penalty becomes the ridge penalty.

#### Objective functions:

For "Gaussian" family is

For the other models are

#### : The shrinkage parameter

The value of has direct relationship with the final result. For instance, in ridge regression, controls the size of the coefficients and amount of regularization.  
If , we could obtain the least squares solutions.  
If , the estimate parameters we get could tend to 0.  
In R, the default value of is a sequence of number. Systems will choose 100 different and the default number of the smallest value of the depends on the sample size relative to the number of the variables. If k> n , the default is 0.0001, on contrast, the default is 0.01.

#### The penalty:

##### LASSO Regularization

when ,the penalty is the lasso penalty,the objective function is

###### Advantage:

Owing to the nature of the -penalty, the lasso could both continuous shrinkage and automatic variable selection simultaneously.

###### Limitation:

* When the k>n, the lasso selects at most n variables before it saturates, because of the convex optimization problem. Moreover, the lasso is not well defines unless the bound on the -norm of the coefficients is smaller than a certain value.
* If there is a group of variables among which the pairwise correlations are very high, then the lasso tends to select only one variable form the group and does not care which one is selected.

##### Ridge penalty

When ,the penalty is the ridge penalty,the function is

###### Advantage:

As a continuous shrinkage method, ridge regression achieves its better prediction performance through a bias- variance trade off.

###### Limitation:

Since it always keeps all the predictors in the model, this regression method could not produce a parsimonious model.

##### Elastic net

The penalty of elastin net is

which is a convex combination of the lasso and ridge penalty.  
The elastic net simultaneously does automatic variable selection and continuous shrinkage, and it can select groups of correlated variables.  
The objective function turns to

**Lemma**: Given data set (y,X) and, define an artificial data set by

Let and . The elastic net criterion can be written as

According to the lemma, this function could be transformed into an equivalent LASSO problem and then be solved.

#### Alpha

Since the value of alpha has directly relationship with the penalty, we choose different value of alpha to expand this algorithm. The default value of alpha is 1, but we apply 0, 0.25, 0.5, 0.75 and 1 to configure different penalty of this algorithm.

(see Zou & Hastie, 2005 for details)

### Classification and regression with Random Forest (Random Forest)

#### Procedures of Constructing Random forest

In Random Forest, a tree is a classifier and the purpose of this algorithm is using different sub data set to construct a number of trees and form a forest. After a large number of trees are generated, they vote for the most popular class for an input vector and give prediction.  
For instance, when generate the tree, a random vector is generated, which is independent of the past random vectors but with the same distribution. And a tree is grown using the training set and , resulting in a classifier , where x is an input vector.Then each tree casts a unit vote for the most popular class at input x.

#### Advantage:

* Random forests do not overfit as more trees as added.
* Since each tress is constructed independently, random forests have lower correlation between classifiers.

#### Some parameters used to expand this algorithm

**ntree**: the number of trees to grow. This should not be set to too small a number to ensure that every input row gets predicted at least a few times. Significant improvements in classification accuracy have resulted from growing an ensemble of trees and letting them vote for the most popular class.  
**mtry**: the number of variables randomly sampled as candidates at each split. The default values are different for classification ( )and regression().  
**maxnode**: maximum size of terminal nodes. Setting this number larger causes smaller trees to be grown and thus take less time. The default values for classification is 1 and for regression is 5.

(see Breiman, 2001 for details)

### Support vector machines (SVM)

svm is used to train a support vector machine. It can be used to carry out general regression and classification, as well as density- estimation.

#### Description

##### Kernel function

1. In classification, SVM separate the different classes of data by a hyper-plane

is a mapping of the inputting data into a high-dimensional feature space.  
Hence the decision function

can be used to represent the result of the classification.  
suppose we have 2 classes, f(x)=1, when x belong to one class OR f(x)=-1, when otherwise.  
It can be shown that the optimal hyper-plane is the one with the maximal margin of the separation between the two classes.

2.**Maximum margin classifier** We suppose the hyper-lane could be represents as , hence the distance from a point to this lane is . The geometrical margin could be represent as and the objective function becomes to

We could simplify this function to

since our objective is to find the parameters w and b.

This function could be transferred to convex quadratic programming as followed

Considering the outliers, we could revise this formula to

Adding Lagrange Duality,

This problem becomes to

The dual form of this formulation is

3. Solve dual form of the formulation  
Through derivation and SMO, we could transfer this formula to

This is C-SVM.

#### Kernel function

A kernel function return the inner product between two points in a suitable feature space, thus defining a notion of similarity, with little computational cost even in very high dimensional spaces.  
The four kernel functions used in R is shown as followed.

|  |  |
| --- | --- |
| *Linear Kernel* |  |
| *Polynomial Kernel* |  |
| *Radial Basis Kernel* |  |
| *Sigmoid Kernel* | ) |

#### Advantages:

Since the quadratic programming problem and the final decision function depend only on dot products between patterns, with the help of kernel function, the inside dot product can be represented by a kernel function k,

we could practically work in space of any dimension without any significant additional cost, as the "kernel trick".  
 In the meantime, with kernel function, SVM could generalize the linear algorithm to the non-linear cases.  
 When deal with multi-class classification, SVM applies one-against-one method. Although this suggests a higher number of support vector machines to train the overall CPU time used id less compared to the onr-against-all method since the problems are smaller and the SVM optimization problem scales super-linearly.

#### Model component

The default setting for the SCM is c-classification or eps-regression, depending on whether the response variable is a factor or not.

#### C-classification

The dual form of the bound constraint C-SVM formulation is:

#### Some parameters used to expand this algorithm

**Type**: svm can be used as a classification machine, as a regression machine, or for novelty detection. Depending of whether y is a factor or not, the default setting for type is C-classification or eps-regression, respectively, but may be overwritten by setting an explicit value.  
Valid options are:  
\* C-classification  
\* nu-classification  
\* one-classification (for novelty detection)  
\* eps-regression  
\* nu-regression

**Degree**: the parameter needed for kernel of type polynomial. The default value of this is 3.

**Gamma**: the parameter needed for all kernels except linear. The default value of this is dimension.

**Coef0**: parameter needed for kernels of type polynomial and sigmoid. The default value of this is 0.

**Cost**: cost of constraints violation. The default value of this is 1. A high cost value C will force the SVM to create a complex enough prediction function to misclassify as few training points as possible, while a lower cost parameter will lead to simpler prediction function.

**Nu**: since in R, the type could transfer to nu-classification as well, we could change this value.  
For nu-classification, the dual formulation becomes

the ν parameter has the interesting property of being an upper bound on the training error and a lower bound on the fraction of support vectors found in the data set, thus controlling the complexity of the classification function build by the SVM.

**Chang, C. C., & Lin, C. J. (2011). LIBSVM: A Library for Support Vector Machines. Acm Transactions on Intelligent Systems and Technology, 2(3).** [**doi:Artn**](doi:Artn) **27 10.1145/1961189.1961199**  
(see Karatzoglou, Meyer, & Hornik, 2006 for details)

### Machine learning with Bayesian additive regression trees(BARTMACHINE)

BART is a Bayesian approach to nonparametric function estimation using regression trees. BART can be considered a sum-of-trees ensemble, with a novel estimation approach relying on a fully Bayesian probability model.  
As a Bayesian model, BART consists of a set of priors for the structure and the leaf parameters and a likelihood for data in the terminal nodes. The aim of the priors is to provide regularization, preventing any single regression tress from dominating the total fit. BARTMACHINE introduces many new features for data analysis using BART.

### Model component:

where Y is the vector of responses, X is the matrix (the predictors column-joined) and is the vector of noise. M is the number of the distinct regression trees, each composed of a tree structure, denoted by T, and the parameters at the terminal nodes, denoted by M. Hence represents an entire tree with both its structure and set of leaf parameters. The set of the tree’s leaf parameters is denoted as where is the number of terminal nodes for a given tree. The observation’s predicted value is the sum of the m leaf values arrived at by recursing down all m trees.

#### Priors and likelihood

The prior for the BART model has three components:  
\* The tree structure itself  
\* The leaf parameters given the tree structur  
\* The error variance which is independent for the tree structure and leaf parameters  
Assume that the conditional independence of the leaf parameters given the tree’s structure.

:This prior component affects the locations of nodes within the trees.  
**Node depth** represents the distance from the root. Nodes at depth d are nonterminal with prior probability ,where ,and . This component of the tree structure prior aims to enforce shallow tree structures, thereby limiting complexity of any single tree and resulting in more model regularization. The default number for this is ,.  
For nonterminal nodes, unlike original formulation, where each available predictor has equal probability to be selected, in BART, this is relaxed to allow for a **generalized Bernoulli distribution**, where the user could specifies , where each denotes the probability of the jth variable being selected a priori.

: This prior component controls the leaf parameters. When giving the set of terminal nodes, each terminal node has a leaf parameter representing the "best guess" of the response in this partition of predictor space. The prior on each of the leaf parameters is given as:   
. The variance is chosen in order to make the range center plus or minus k variance cover 95% of the provided response values in the training set. The default value of k is 2. The aim of this prior is to provide model regularization by shrinking the leaf parameters toward the center of the distribution of the response. The larger the value of k, the smaller the value of , resulting in more model regularization.

:this component of prior is on the error variance and is chosen to be .

The default values of has good performance, but optimal tuning can be achieved via cross-validation. Along with a set of priors, BART specifies the likelihood of responses in the terminal nodes. They are assumed a priori normal with the mean being the "best guess" in the leaf at the moment and variance being the best guess of the variance at the moment,

#### Posterior distribution and prediction

A key feature of BART is to employ a form of "Bayesian backfitting", where the jth tree is fit iteratively, holding all other m-1 trees constant by exposing only the residual response that remains unfitted:

The procedure is shown as followed:

First, proposing a change to the first tree’s structure T, which is accepted or rejected via a Metropolis-Hastings step, since the posterior of the tree structure does not depend on the leaf parameters. The revising include small perturbations to the tree structure: growing a terminal node by adding two child nodes, pruning two child nodes, or changing a split rule.

Given the tree structure, the posterior then revise the leaf parameters. This procedure progresses iteratively for each tree, using the updated set of partial residuals . The posterior of each of the leaf parameters in M is conjugate normal with its mean being a weighted combination of the likelihood and prior parameters.

Finally, conditional on the updates set of tree structure and leaf parameters, a draw from the posterior of is made based on the full model residuals

All these steps represent a single Gibbs iteration. Generally, no more than 1,000 iterations are needed as burn-in(the default value of burn-in node is 250). a single predicted value f(x) can be obtained by taking the average of the posterior values and a quantile estimate can be obtained by computing the appropriate quantile of the posterior values.

#### Advantages:

* BARTMACHINE is fully parallelized during the model creation, prediction, and many of the other features.
* The BARTMACHINE package also implements the variable selection procedures, which are best applied to data problems where the number of covariates influencing the response is small relative to the total number of covariate.

##### Some parameters used to expand this algorithm

Like RandomForest algorithm, the main parameters we could used to expand this algorithm is  
\* num\_trees: the number of trees to be grown in the sum-of-trees model.

(see Kapelner & Bleich, 2016 for details)

## SuperLearner Loop

The super learner function takes a training set pair (X,Y) and returns the predicted values based on a validation set. With the V-fold cross validation theorem, this function could extent the candidate learner selector to include weighted averages of the candidate learners.

### Algorithm:

Step 0: train each candidate learner on entire dataset.  
Step 1: Split data into v blocks.  
Step 2: train each candidate learner.  
Step 3: predict the outcomes in the validation block based on the corresponding training block candidate learner.  
Step 4: model selection and fitting for the regression of the observed outcome onto the predicted outcomes from the candidate learners.

is the coefficients of each candidate learners and z represents learners.  
Step 5: evaluate super learner by combining predictions from each candidate learner with m(z;).

Usually, the default method on estimating the coefficients for the super learner and the model to combine the individual algorithms in the library is NNLS.

#### NNLS (The Lawson-Hanson algorithm for non-negative least squares):

y solving the problem

In fact, by using super learner, we could predict m subjects by n methods and b is the real value of this m subjects. With NNLS, we could find optimal x (the coefficients of different algorithms), which lead to min error.  
In R, there always has a prescreen algorithms as well, which first rank the variables in X based on either a univeriate regression p-value of the randomForest variable importance. A subset of the variables in X is selected based on a pre-defined cut-off. With this subset of the X variables, the algorithm in SL.Library are then fit in.

#### Value

* call: the configuration of this superlearner.
* libraryNames: a character vector with the names of the algorithmn in the library, including the risk and coefficient for them.
* SL.library: returns in the same format as SL.librart in the argument.  
  This will help us to have a clear mind about which algorithms are included in this superlearner function.
* SL.predict:the predict value from the super learner.
* coef: the coefficients for all the algorithms including in the SL.library.  
  With the help of **coef**, we could compare the parameters(weighted) of each algorithm in the SL.library when obtaining the final super learner predicted value. Undoubtedly, the algorithm, whose parameter is larger than others, plays a more important role in predicting procedure, which means that this algorithm has fit better for this data set than others.
* library.predict:a matrix with the predicted values from each algorithm in the SL.library.
* Z: a matrix with the cross-validation prdicted values from each algorithm in the SL.library.
* cvRisk: a numeric vector with the v-fold cross-validated risk estimate for each algorithm in the SL.library without the the CV risk estimate for the superlearner.
* family: return the information about the family and link function. Through this, we could check different variables' family and, maybe , make further improvements by changing their link functions.
* futLibrary: a list with the fitted objects for each algorithm in SL.library on the full training data set.
* varNames: a character vector with the names of the variables in X.
* validRows: a list containing the row numbers for the v-fold cross-validation step.
* method: the method that used to combine algorithms in the SL.library to get superlearner prediction and compute error.We could change different methods and compare their computed error to get better prediction.
* whichScreen: a logical matrix indicating which variables passed each screening algorithm.
* Control: the control list.
* cvControl: the cvControl list.
* errorInCVLibrary: A logical vector indicating if any algorithms experienced an error within the CV step.
* errorsInLibrary: A logical vector indicating if any algorithms experienced an error on the full data.

(see 2002 for details)

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. Journal Article. [http://doi.org/Doi 10.1023/A:1010933404324](http://doi.org/Doi%2010.1023/A:1010933404324)

Fox, J., Weisberg, S., & Fox, J. (2011). *An r companion to applied regression* (2nd ed., p. xxii, 449p.). Book, Thousand Oaks, Calif.: SAGE Publications.

Hastie, T., & Tibshirani, R. (1999). *Generalized additive models* (p. xv, 335p.). Book, Boca Raton, Fla.: Chapman & Hall/CRC. Retrieved from [Publisher description http://www.loc.gov/catdir/enhancements/fy0916/99015185-d.html](Publisher%20description%20http://www.loc.gov/catdir/enhancements/fy0916/99015185-d.html)

Kapelner, A., & Bleich, J. (2016). BartMachine: Machine learning with bayesian additive regression trees. *Journal of Statistical Software*, *70*(4). Journal Article.

Karatzoglou, A., Meyer, D., & Hornik, K. (2006). Support vector machines in r. *Journal of Statistical Software*, *15*(9). Journal Article. Retrieved from [<Go to ISI>://WOS:000236800500001](%3cGo%20to%20ISI%3e://WOS:000236800500001)

Venables, W. N., Ripley, B. D., & Venables, W. N. (2002). *Modern applied statistics with s* (4th ed., p. xi, 495p.). Book, New York: Springer. Retrieved from [Table of contents http://www.loc.gov/catdir/toc/fy042/2002022925.html Publisher description http://www.loc.gov/catdir/enhancements/fy0812/2002022925-d.html](Table%20of%20contents%20http://www.loc.gov/catdir/toc/fy042/2002022925.html%20Publisher%20description%20http://www.loc.gov/catdir/enhancements/fy0812/2002022925-d.html)

Wood, S. N. (2006). *Generalized additive models : An introduction with r* (p. xvii, 391p.). Book, Boca Raton, FL: Chapman & Hall/CRC. Retrieved from [Publisher description http://www.loc.gov/catdir/enhancements/fy0702/2006040209-d.html](Publisher%20description%20http://www.loc.gov/catdir/enhancements/fy0702/2006040209-d.html)

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net (vol b 67, pg 301, 2005). *Journal of the Royal Statistical Society Series B-Statistical Methodology*, *67*, 768–768. Journal Article. [http://doi.org/DOI 10.1111/j.1467-9868.2005.00527.x](http://doi.org/DOI%2010.1111/j.1467-9868.2005.00527.x)

(2002). Generic.