Short descriptions of included learners

Generalized Linear Model

SL.glm() fits a generalized linear model (GLM) with a binomial variance function ($\mu(1-\mu)/n$) and logit link function ($\log\left(\frac{\mu}{1-\mu}\right)$). This is more simply described as a logistic regression where only the main effects (no interactions) were entered into the model.

Penalized regression model using elastic net

SL.glmnet() fits a GLM with elastic net regularization via maximization of a penalized binomial log-likelihood (or a regularized logistic regression). Specifically, the default implementation used here sets $\alpha = 1$ (equivalent to the "least absolute shrinkage and selection operator" or LASSO penalty), The lasso is often used to fit a more parsimonious linear model, since it "shrinks" the coefficient estimates – usually shrinking some coefficients to zero, depending on the regularization parameter setting used. The optimal regularization parameter (λ) was determined via minimization of a 10-fold cross-validated binomial deviance loss function.

Generalized Additive Model

SL.gam() fits a generalized additive model (GAM). Specifically, in the default implementation used here, each continuous predictor (containing at least 5 unique values) is modeled as a nonparametric smoothing term – a smoothing spline with two degrees of freedom that is a function of only the original predictor. These smoothing splines smooth "indirectly" and utilize penalized least squares, whereas the overall GAM is fit iteratively using a backfitting algorithm. If any discrete predictors (containing fewer than 5 unique values) are included, these are not smoothed but are entered into the model as-is. The error distribution is assumed to be binomial and the logit link function is used. A maximum of 50 local scoring iterations and 50 backfitting iterations are allowed.

Breiman's random forest algorithm

As used in this context, SL.randomForest() estimates a bootstrap aggregated (bagged) ensemble of random decision trees for classification. A thousand trees are grown, and there is no limit imposed on either the number of observations contained in the terminal nodes of each tree (meaning these can contain as few as a single observation) or the number of terminal nodes the trees can have (meaning trees are grown to the maximum possible size). At each split, the number of predictors randomly sampled as candidates is the square root of the total number of predictors available. The predicted outcome is the proportion of out-of-bag (OOB) 'votes' for the larger value of the binary outcome variable.

Extreme Gradient Boosting machine

SL.xgboost() fits a gradient boosted ensemble of decision trees. The classification task outputs predicted outcome in the form of probabilities. A thousand iterations are allowed. Trees are limited to a depth of four and each terminal node must contain at least ten observations. Each tree's contribution is divided by ten when added to the ensemble. These three constraints (depth of trees, size of nodes, and learning rate) are intended to protect against overfitting and result in an ensemble that changes at a slower rate and contains more "stump"-like trees than, for example, the hyperparameter settings for randomForest above.