"Stacking" for SEL and *0-1* Loss

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

Linear combinations of SEL predictors

What might be suggested in the spirit of combining predictors, but without a Bayesian flavor? Suppose that M predictors are available (all based on the same training data), $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_M$. Under squared error loss, one might seek a (weight) vector $\mathbf{w}$ for which the predictor

$$\hat{f}(\mathbf{x}) = \sum_{m=1}^{M} w_m \hat{f}_m(\mathbf{x})$$

is effective. As this linear form is inherently more flexible than any one of its constituent predictor forms, it potentially provides important reduction of model bias and improved overall prediction.

Optimal weight vector

Ideally, one might hope to approximate

$$\widehat{\mathbf{w}} = \arg\min_{\mathbf{w}} \mathsf{E}^{\mathsf{T}} \mathsf{E}^{(\mathbf{x},y)} \left(y - \sum_{m=1}^{M} w_m \hat{f}_m (\mathbf{x}) \right)^2$$

That this could improve on any single one of the $\hat{f}_m$s is in some sense "obvious," since the $\mathbf{w}$ over which the minimization is done includes vectors with one entry 1 and all others 0.

Theoretical optimizer

Consider the random vector $\left(\hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), \ldots, \hat{f}_M(\mathbf{x}), y\right)' = \left(\hat{\mathbf{f}}', y\right)$. Let

$$\underset{M\times M}{\mathrm{E}\left(\hat{\mathbf{f}}\hat{\mathbf{f}}'\right)} \quad \text{and} \quad \underset{M\times 1}{\mathrm{E}y\hat{\mathbf{f}}}$$

be respectively the matrix of expected products of the predictions and vector of expected products of y and elements of $\hat{\mathbf{f}}$ (under the the $P^N \times P$ joint distribution of $(\mathbf{T}, (\mathbf{x}, y))$). Upon writing out the expected square to be minimized and doing some matrix calculus, it's possible to see that theoretically optimal weights are of the form

$$\widehat{\mathbf{w}} = \left(\mathrm{E}\left(\hat{\mathbf{f}}\hat{\mathbf{f}}'\right)\right)^{-1} \mathrm{E}y\hat{\mathbf{f}}$$

Of course, this is not usable in practice, as typically the mean vector and expected cross product matrix are unknown.

"LOOCV winner" weight vector

What can be done in practice is (for $\hat{f}_m^i$ the mth predictor fit to $\mathbf{T} - \{(\mathbf{x}_i, y_i)\}$, the training set with case i deleted) to look for a LOOCV "winner" weight vector

$$\mathbf{w}^{\text{stack}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{N} \left(y_i - \sum_{m=1}^{M} w_m \hat{f}_m^i (\mathbf{x}_i) \right)^2$$

and then ultimately use the "stacked" predictor of y

$$\hat{f}(\mathbf{x}) = \sum_{m=1}^{M} w_m^{\text{stack}} \hat{f}_m (\mathbf{x})$$

Ad hoc SEL stacking

An ad hoc version of stacking-type averaging that (because of computational load) is probably more common than the careful approach outlined on the previous slide, is choice of weight vector $\mathbf{w}^{\text{stack}}$ based on informal consideration of one's (CV-supported) evaluation of the effectiveness of the individual predictors $\hat{f}_m(\mathbf{x})$ and the (training set) correlations between them. (Averaging multiple highly correlated predictors can't be expected to be particularly helpful, and individually effective predictors should get more weight than those that are relatively speaking ineffective.)

Ad hoc stacking of *0-1* loss classifiers

The application of the general notion of stacking to 0-1 loss classification has typically been treated on a very informal and ultimately unprincipled basis. Probably the most common suggestion extant in the machine learning world is to make classifications on a "majority vote" of an ensemble of classifiers. This is completely unsupported by any sensible theory. In this regard, see Vardeman and Morris "Majority Voting by Independent Classifiers can Increase Error Rates" that appears in *The American Statistician* in 2013 and their "Reply" to comments on the paper by Baker and others that appeared in the same journal in 2014.

Principled stacking of *0-1* loss classifiers

A principled line of reasoning for the classification case is this. If a 0-1 loss classifier is any good, it approximates the optimal form. So if it has an underlying voting function, that must be equivalent to (must be a monotone transform of) an *approximate likelihood ratio*. What one is doing is attempting to make a better approximate likelihood ratio by combining several of these. It is then sensible to use underlying voting functions for the classifier (and the classifiers themselves in cases where no such voting function is available) **as features** input into a tree-based classification methodology.

Tree-based stacking of classifiers

A tree-based methodology is appropriate because of invariance to monotone transformation of coordinates of inputs and the fact that constituent voting functions are potentially on completely different scales, e.g. in some cases involving approximations for linear functions of $P[y = 1|\mathbf{x}]$ and in others approximations for $\mathcal{L}(\mathbf{x})$ directly Details of sensible cross-validation to choose parameters of the constituent classification methods and the final tree-based method in this context remain to be considered. But the basic approach is clear and principled.