Generalized Stacking and "Deep" Structures

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

Combinations of “ensembles” of predictors

Suppose $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_M$ (all based on the same training data) are available. We might call them an "ensemble" of predictors and hope to make from them a single predictor that is more effective than any of the constituents. For SEL prediction, a linear combination of these is one way of making such a predictor. For 0-1 loss classification, combining multiple classifiers through a tree-based function of their voting functions seems likely to be generally practically effective. (There is, unfortunately, a large and very confused "theoretical" literature on "classifier fusion" mostly built around the ad hoc notion of combination via majority voting.) Here we consider the general problem "predictor combination." The primary contribution it potentially offers is **reduction of model bias** by adding flexibility not provided by any individual $\hat{f}_m$.

Generalized stacking/super-learners/meta-predictors

One important way to view the stacked SEL predictor

$$\hat{f}(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \hat{f}_m(\mathbf{x})$$

is as a *linear* predictor **based on M new "features"** that are the values of the ensemble. This suggests applying *any* good predictor methodology to a "training set" consisting of M vectors of predictions ... *with or without some or all of the original input variables also reused as inputs.* This generalization of ordinary stacking is

$$\tilde{f}(\mathbf{x}) = \hat{f}\left(\hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), \ldots, \hat{f}_M(\mathbf{x}), \mathbf{x}\right) \qquad (1)$$

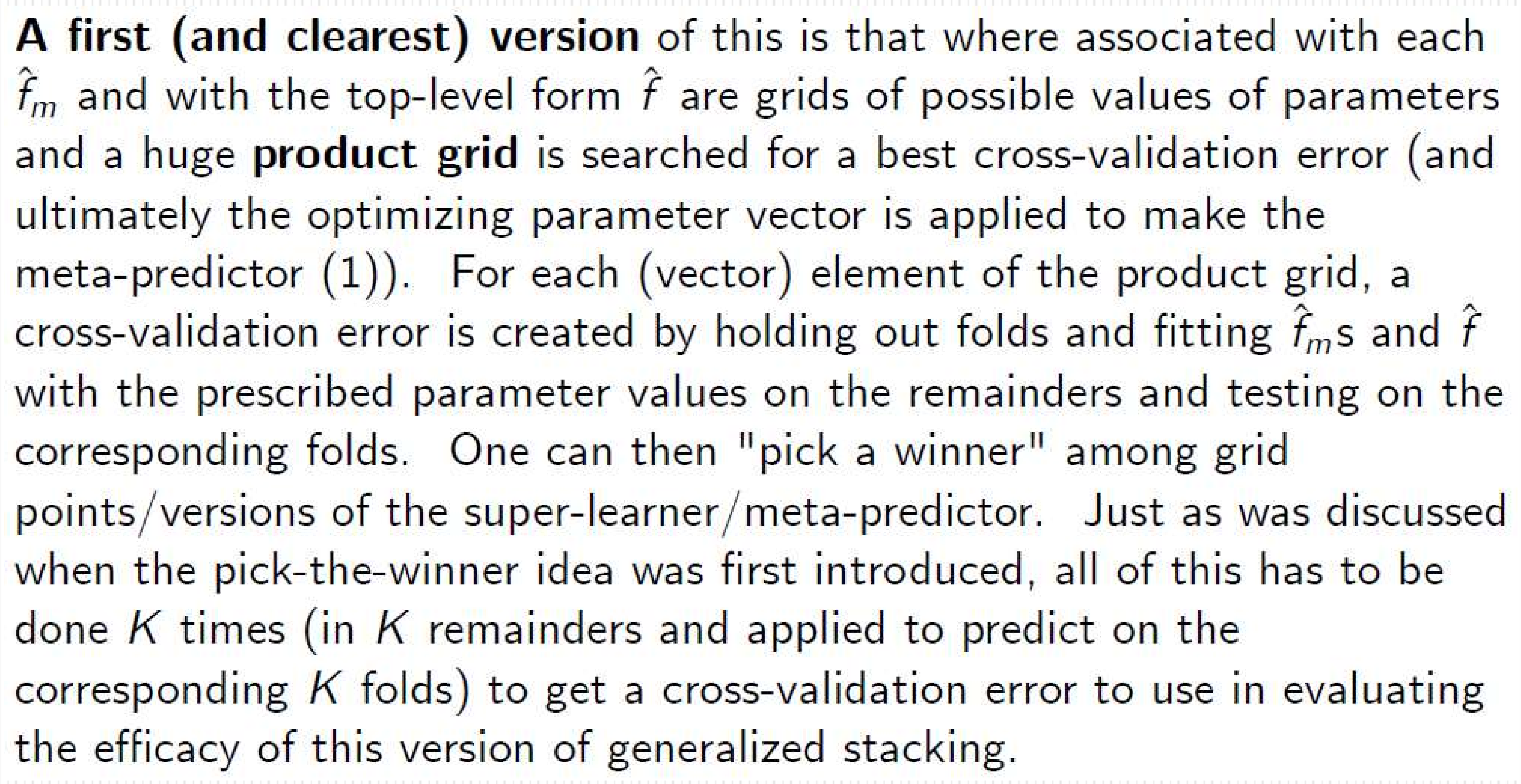
for some appropriate form $\hat{f}$. This is more general than ordinary stacking, applicable beyond SEL, and has the potential to be even more effective than a linear combination of the M predictors in SEL

Perspective on generalized stacking

(Generalized) Stacking is a big deal. From the earliest of the public predictive analytics contests (the Netflix Prize contest run 2006-2009) it has been common for winning predictions to be made by "end-of-game" merging of effort by two or more separate teams that in some way average their separate predictions. More and more references are made on contest forums to various strategies for combining basic predictors. Multiple-level versions of the stacking structure are even discussed (though in truth, they are but structured versions of the general form (1)).

While the success of some (?luckiest among a number of?) ad hoc choices of generalized stacking forms in particular situations is undeniable, principled choices of forms and parameters for $\hat{f}$ (and indeed $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_M$) in (1) involve both logical subtleties and huge computational demands.

A first version of parameter choice



A first (and clearest) version of this is that where associated with each $\hat{f}_m$ and with the top-level form $\hat{f}$ are grids of possible values of parameters and a huge **product grid** is searched for a best cross-validation error (and ultimately the optimizing parameter vector is applied to make the meta-predictor (1)). For each (vector) element of the product grid, a cross-validation error is created by holding out folds and fitting $\hat{f}_m$s and $\hat{f}$ with the prescribed parameter values on the remainders and testing on the corresponding folds. One can then "pick a winner" among grid points/versions of the super-learner/meta-predictor. Just as was discussed when the pick-the-winner idea was first introduced, all of this has to be done K times (in K remainders and applied to predict on the corresponding K folds) to get a cross-validation error to use in evaluating the efficacy of this version of generalized stacking.

A second version of parameter choice

A second version of generalized stacking might pertain where individually-"optimized" (by cross-validation or other means) versions of the $\hat{f}_m$s will be combined into a form (1) and choice of complexity parameters for $\hat{f}$ then made by applying another subsequent "cross-validation" treating the chosen forms for the $\hat{f}_m$ as fixed. This is logically a complicated process that is different from the earlier one. The only way to assess its potential performance is to do it (K times) on K remainders to make predictions on the corresponding folds and ultimately produce a cross-validation error. That is, within each of K remainders the whole sequence of choosing parameters for the $\hat{f}_m$s and subsequently for the $\hat{f}$ must be repeated (by making K folds and remainders **within each remainder** ... surely leading to different "best" vectors of parameters for each fold) and applied to the corresponding fold to get a reliable cross-validation error.

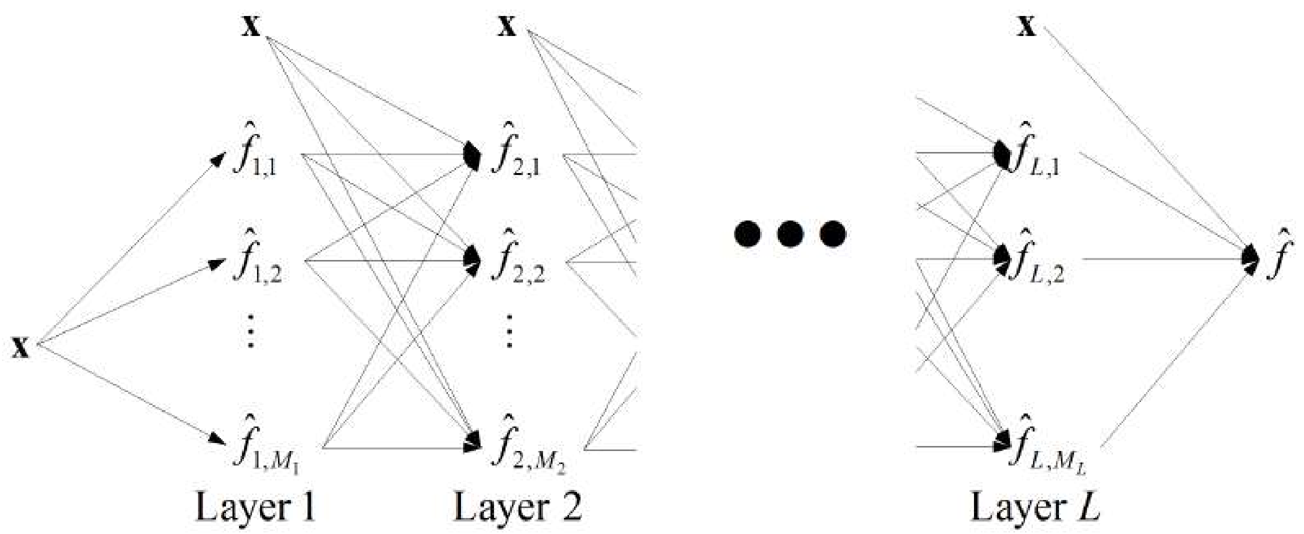
Computational burden; top level form

It is clear then that computation grows rapidly with the complexity of constituent predictor forms, the breath of the optimization desired, and the extent to which repetition of cross-validation is used in both making and testing the efficacy of a super-learner.

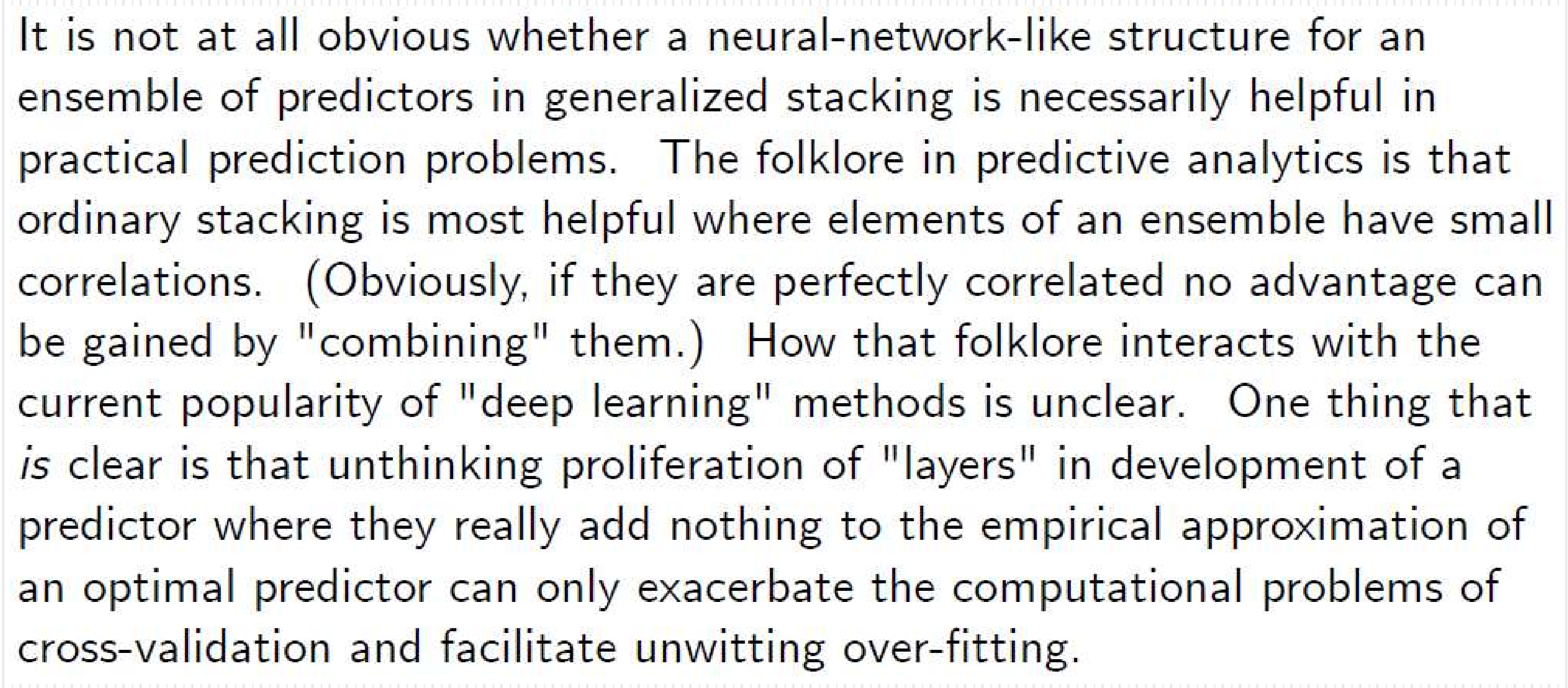
What kind of top-level $\hat{f}$ should be used in predictor (1) can in theory be investigated by comparison of cross-validation errors. The linear form is most common and (at least in its ad hoc application) famously successful. But there is a case to be made that a random forest form has potential to be at least as effective in this role. Its invariance to scale of predictors (inherited from its tree-based construction) and wide success and reputation as an all-purpose tool make it a natural candidate.

Deep structures in generalized stacking

Neural networks have the kind of "(potentially repeated) composition of multiple functions of the input vector" character evident in form (1). That motivates consideration of generalized stacking where the ensemble of predictors $\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_M$ itself has some specific kind of "neural-network-like" structure behind it. Below is a graphical representation of what is possible.



Multiple layers in generalized stacking



It is not at all obvious whether a neural-network-like structure for an ensemble of predictors in generalized stacking is necessarily helpful in practical prediction problems. The folklore in predictive analytics is that ordinary stacking is most helpful where elements of an ensemble have small correlations. (Obviously, if they are perfectly correlated no advantage can be gained by "combining" them.) How that folklore interacts with the current popularity of "deep learning" methods is unclear. One thing that *is* clear is that unthinking proliferation of "layers" in development of a predictor where they really add nothing to the empirical approximation of an optimal predictor can only exacerbate the computational problems of cross-validation and facilitate unwitting over-fitting.