Supplementary Materials for "Stacked Regression and Poststratification"

Joseph T. Ornstein

November 4, 2019

Appendix A Synthetic Poststratification Proof

The following proof demonstrates that synthetic poststratification and classical MRP produce identical estimates if the first-stage model is additively-separable. Let $\hat{\mathbf{y}}$ be the vector of predictions for each type of respondent, and $\mathbf{p}$ be the true empirical pmf for each type. The classical MRP poststratified estimate is the dot-product $\hat{\mathbf{y}} \cdot \mathbf{p}$. MrsP uses the same vector of predictions $\hat{\mathbf{y}}$, but uses a synthetic joint probability distribution, where each entry is the product of marginal probabilities. I will denote this synthetic poststratification vector as $\boldsymbol{\pi}$. Therefore, the poststratified MrsP estimates will be $\hat{\mathbf{y}} \cdot \boldsymbol{\pi}$.

Let X_1 through X_m be discrete random variables, and the $c \times m$ matrix X be a matrix in which the each row is one of the c possible combinations of values that X_1 through X_m can take. Crucially, we are not assuming that X_1 through X_m are independent, so $P(X_1 = x_{1i}, ..., X_m = x_{mk})$ need not equal $P(X_1 = x_{1i})...P(X_m = x_{mk})$.

Suppose the model is additively separable, such that $\hat{y} = X\hat{\beta}$. The vector of MrsP predictions for each unit is therefore $\pi'\hat{y}$, where π is the synthetic distribution vector. To complete the proof, we must show that $p'X\hat{\beta} = \pi'X\hat{\beta}$. Because β is a vector, this is equivalent to showing that $p'X = \pi'X$:

$$p'X = \begin{bmatrix} \sum_i ... \sum_k P(X_1 = x_{1i}, ..., X_m = x_{mk}) x_{1i} \\ \vdots \\ \sum_i ... \sum_k P(X_1 = x_{1i}, ..., X_m = x_{mk}) x_{mk} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_i P(X_1 = x_{1i}) x_{1i} \\ \vdots \\ \sum_k P(X_m = x_{mk}) x_{mk} \end{bmatrix}$$

$$= \begin{bmatrix} \sum_i ... \sum_k P(X_1 = x_{1i}) ... P(X_m = x_{mk}) x_{1i} \\ \vdots \\ \sum_i ... \sum_k P(X_1 = x_{1i}) ... P(X_m = x_{mk}) x_{mk} \end{bmatrix} = \pi' X$$

This completes the proof. If our underlying first-stage model is additively separable, then our poststratified estimates will be identical whether we use MrsP or classical MRP.

Appendix B First Stage Models

B.1 LASSO

The LASSO (least absolute shrinkage and selection operator) is an approach to regularization and variable selection for regression analysis first popularized by Tibshirani (1996). Whereas ordinary least squares estimates parameters by minimizing the sum of squared residuals, LASSO regression minimizes a *penalized* sum of squared residuals. This penalty term takes the form $\lambda \sum |\beta_j|$, which grows larger with the magnitude of the β coefficients. This penalty "shrinks" the parameter estimates towards zero, ensuring that the resulting model balances fit and complexity. A more complex model, with many nonzero parameter estimates, might more accurately predict the training data, but it is more prone to overfitting. The λ term can be selected through cross-validation, choosing the value that yields the best out-of-sample predictive error.

As Tibshirani (1996) notes, the LASSO estimate has a Bayesian interpretation as well: the LASSO estimate is equivalent to estimating parameters through Bayesian regression, with a prior distribution heavily weighted towards zero. This explains why the LASSO shrinks many parameter estimates to zero, making the resulting model both less prone to overfitting and more interpretable.

B.2 Random Forests

The random forest, first introduced by Breiman (2001), is itself an ensemble approach to classification and regression. Rather than estimating a single model, the procedure constructs a large collection of models, then aggregates their predictions together. Each component model is a regression tree, a technique that generates predictions by successively partitioning the data on the X variables and taking the average outcome of observations at each terminal

node.[1] To ensure that these trees are not all identical, each tree is trained on a bootstrap sample of the dataset (thus the "random" in random forest). The forest prediction is then equal to the mean prediction of the constituent trees. This process is also called "bootstrap aggregation", or "bagging".

When adapting random forest to estimate probabilities for poststratification, we must make a few adjustments. Random forests tend to perform well at classification of binary variables, fully growing a large set of trees from bootstrapped samples and predicting the class based on majority vote. Taking the average prediction from these trees, however, is not a principled method for estimating probabilities; there is no guarantee that the percentage of trees predicting a value of 1 is a well-calibrated estimate of the true probability (Olson and Wyner, 2018). Instead, I use a variant of random forest proposed by Malley et al. (2012). The probability forest algorithm requires the terminal nodes in each tree to contain a large number of observations. This ensures that each terminal node produces probability estimate rather than a binary classification, which can then be averaged across trees. Malley et al. (2012) recommends setting the minimum node size to 10% of the training set size, but for the SRP procedure I tune this hyperparameter to minimize cross-validated Log Loss (see Appendix C for details).

B.3 Gradient Boosting

Like random forests, the gradient boosting method (GBM) is an approach to prediction using ensembles of trees. Unlike random forest, however, the component trees are not grown simultaneously, but sequentially. Each successive tree is built with the goal of improving the overall fit of the ensemble; if there are some observations in the training set that the current ensemble predicts poorly, then the new tree will be trained to better predict those

[1] See Montgomery and Olivella (2018) for an excellent introduction to tree models in political science research. One section discusses using regression trees to improve MRP estimates, an insight that this paper expands.

observations. In practice, this is accomplished by fitting each tree to the negative gradient of the loss function. An intuitive way to think about this process is that each tree is fit to the residuals of the previous prediction, rather than the outcomes. See Friedman (2001) for technical details and Montgomery and Olivella (2018) for more applications on gradient boosting in political science research.

Unlike random forest, GBM requires careful tuning in order to avoid overfitting. In particular, there are two hyperparameters that must be chosen in advance by the researcher. The parameter B governs the maximum number of splits that can be made in each tree. When B is small, each constituent tree will model less complex interaction effects. The parameter ν is a "shrinkage" term, which scales the contribution of each new tree to the overall prediction. If the prediction of tree m is denoted T_m, then the ensemble prediction after growing the m^{th} tree is $f_m = f_(m-1) + \nu T_m$. Smaller values of ν typically yield better predictions (because the ensemble learns across a larger set of trees), but it takes longer for the algorithm to converge. Commonly, researchers will select the values of B and ν through cross-validation, and select an optimal number of trees at the point where prediction error on the test set stops improving. This is the tuning procedure that I implement for SRP.

B.4 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is an intuitive nonparametric approach to regression and classification. For each observation i, KNN predicts an outcome $\hat{y}_i$ by taking the k most similar observations in the training data (according to some predefined distance metric) and computing the mean of their observed outcomes. In classical KNN, this is an unweighted average of the k-nearest neighbors, but a more general approach uses a weighted average, with weights proportional to inverse distance. For the SRP procedure, I use the weighting function proposed by (Samworth, 2012).

As with random forests, the researcher need not assume a model of the DGP in order to

produce estimates. Instead, KNN requires a more easily-accepted assumption: that similar people who live in similar places are likely to hold similar opinions.

Another advantage of this approach is that KNN can easily incorporate spatial predictors. For example, if each survey respondent provides their county of residence, then a prediction using KNN could incorporate the latitude and longitude of that county's centroid as predictors. Predictions would then be generated by a weighted average of nearest neighbors in *physical* space as well as some abstract variable-space. If black respondents in Tennessee have systematically different opinions than black respondents in Minnesota, then the KNN prediction would reflect that, without the researcher having to specify a battery of interaction terms in advance.

Good predictions from KNN depend on selecting a good value for k. Commonly, this value is selected through cross-validation, searching for the value of k that yields the best predictions on a held-out test set. For all of the paper's analyses, I proceed in this fashion, choosing the value of k that minimizes a loss function (RMSE for continuous dependent variables, Log Loss for binary dependent variables) in LOOCV.

B.5 Other Methods

There are several other methods that I considered including in the ensemble, but ultimately chose to omit. One is Bayesian Additive Regression Trees (BART), another tree ensemble technique introduced by Chipman, George and McCulloch (2012). Conceptually, the approach is very similar to GBM; trees are grown sequentially and fit to residuals from the existing ensemble. Unlike GBM, however, each tree's contribution to the ensemble is weighted by a prior distribution, which is updated using Bayesian MCMC. Although the performance of this approach compares favorably to other tree-based models (Montgomery and Olivella, 2018), it tends to produce predictions that are highly correlated with predictions from random forest and gradient boosting. As a result, it does not significantly improve the

performance of an ensemble that already includes the other two methods. Because it is the most computationally-intensive of the three to tune and estimate, I omit BART from the SRP ensemble.

I also omit two other widely-used machine learning classification techniques – Naive Bayes (NB) and Support Vector Machines (SVM) – because they tend to produce poorly calibrated probability estimates without significant modifications (Platt, 1999; Zhang, 2004). Poststratification of binary outcomes requires the fist-stage model to produce a probability for each grid cell, so these methods are a poor fit for our purposes here.

Appendix C Monte Carlo Technical Summary

The X variables are generated by discretizing each Z variable, according to procedure in Table 1. Subnational units are assigned using the Z_4 variable. The N observations with the smallest value of Z_4 are assigned to Unit 1, the next smallest N observations assigned to Unit 2, and so on.

Table 1: Assignment procedure for X variables

Z	X
Less than 1 SD below mean	1
1 SD below mean to mean	2
Mean to 1 SD above mean	3
More than 1 SD above mean	4

The functions D^0 and D^1 in the data-generating process are defined as follows, so that the former is increasing as it approaches (0,0), while the latter is decreasing.

$$D_i^0 = \sqrt{2} - \sqrt{lat_i^2 + lon_i^2}$$

$$D_i^1 = \frac{\sqrt{lat_i^2 + lon_i^2}}{2}$$

Table 2 lists the parameter values swept in the Monte Carlo. All simulation code will be made available at the author's website.

Table 2: List of parameter values used in the Monte Carlo Simulation

Parameter	**Values**	**Description**
ρ	{0.2, 0.4, 0.6}	Correlation between Z variables
θ	{0, 1, 2, 3, 4, 5}	Strength of the threeway interaction effect
n	{3000, 5000, 10000}	Sample size drawn for disaggregation, MRP, and SRP estimates
N	15000	Observations per unit
M	200	Number of units
σ^2	5	Error term variance in DGP

Appendix D Software Details and Implementation

All the computations described in this paper were conducted in `R`. For each machine learning algorithm, Table 3 lists the software package used, the hyperparameters that were tuned, and the parameter values used. For any model that requires parameter tuning, I perform a grid search on all combinations of the parameters listed, selecting the combination that minimizes RMSE (for continuous dependent variables) or Log Loss (for binary dependent variables) in a 5-fold cross-validation. For KNN, I use the built-in LOOCV function from the `kknn` package to tune k.

Table 3: Machine Learning Techniques, Packages, and Parameter Tuning

Technique	Package	Parameters	Values
LASSO	`glmnet`	λ	0.0005 to 0.047
KNN	`kknn`	k	20 to 305
Random Forest	`ranger`	min.node.size	1 to min(501, $\frac{n}{5}$)
Gradient Boosting	`xgboost`	max_depth	$\{1, 2, 3, 4, 6\}$
		min_child_weight	$\{1, 3, 5, 7, 9\}$
		subsample	$0.8, 1$
		colsample_bytree	$0.8, 1$
		num_round	Max $10,000$

References

Breiman, Leo. 2001. "Random forests." *Machine Learning* 45(1):5–32.

Chipman, Hugh A., Edward I. George and Robert E. McCulloch. 2012. "BART: Bayesian additive regression trees." *Annals of Applied Statistics* 6(1):266–298.

Friedman, Jerome H. 2001. "Greedy Function Approximation: A Gradient Boosting Machine." *The Annals of Statistics* 29(5):1189–1232.

Malley, J.D., J. Kruppa, A. Dasgupta, K.G. Malley and A. Ziegler. 2012. "Probability Machines: Consistent Probability Estimation Using Nonparametric Learning Machines." *Methods of Information in Medicine* 51(1):74–81.

Montgomery, Jacob M and Santiago Olivella. 2018. "Tree-Based Models for Political Science Data." *American Journal of Political Science* 62(3):729–744.

Olson, Matthew and Abraham J Wyner. 2018. "Making Sense of Random Forest Probabilities: A Kernel Perspective." *Working Paper* pp. 1–35.

Platt, John C. 1999. "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods." *Advances in large margin classifiers* 10(3):61–74.

Samworth, Richard J. 2012. "Optimal weighted nearest neighbour classifiers." *Annals of Statistics* 40(5):2733–2763.

Tibshirani, Robert. 1996. "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B (Methodological)* 58(1):267–288.

Zhang, Harry. 2004. "The Optimality of Naive Bayes." *American Association for Artificial Intelligence* 1(2):1–6.