Cross-Cluster Weighted Forests

Maya Ramchandran[1], Rajarshi Mukherjee[1], and Giovanni Parmigiani[1, 2]

[1]*Department of Biostatistics, Harvard T.H. Chan School of Public Health*
[2]*Department of Data Sciences, Dana-Farber Cancer Institute, Boston, MA 02115, USA*

Abstract

Adapting machine learning algorithms to better handle the presence of natural clustering or batch effects within training datasets is imperative across a wide variety of biological applications. This article considers the effect of ensembling Random Forest learners trained on clusters within a single dataset with heterogeneity in the distribution of the features. We find that constructing ensembles of forests trained on clusters determined by algorithms such as k-means results in significant improvements in accuracy and generalizability over the traditional Random Forest algorithm. We denote our novel approach as the **Cross-Cluster Weighted Forest**, and examine its robustness to various data-generating scenarios and outcome models. Furthermore, we explore the influence of the data-partitioning and ensemble weighting strategies on conferring the benefits of our method over the existing paradigm. Finally, we apply our approach to cancer molecular profiling and gene expression datasets that are naturally divisible into clusters and illustrate that our approach outperforms classic Random Forest. Code and supplementary material are available at https://github.com/m-ramchandran/cross-cluster.

1 Introduction

Datasets containing natural clusters or batch effects are ubiquitous across most biological applications, necessitating the advent of prediction algorithms that can adapt to the particular challenges of handling possible heterogeneity in the distribution of the features. Numerous learning algorithms have been created to address the setting in which the covariate-outcome relationship varies across clusters, including mixed-effects regression, sequential ensembling approaches, the mixture of experts framework, and dynamic co-clustering learning algorithms [1] [2] [3]. This context is in fact analogous to the multi-study framework formalized by Patil and Parmigiani (2018), in which separate clusters can be thought of as individual studies [4]. Multi-study learning handles the availability of multiple training studies that measure the same outcome and many of the same covariates by building ensembles of learners each trained on a single study to form the final predictor. Several learning algorithms have been shown to be highly effective in this scheme, including regularized regression, neural networks, and Random Forest. In this paper, we extend these results to additionally consider the situation in which the true outcome model remains relatively constant across clusters

and the primary source of heterogeneity is in the distribution of the covariates; this requires prediction algorithms to now be able to disentangle the true covariate-outcome relationship from variation introduced by the presence of clusters. We particularly focus on ensembles of Random Forests, in which each forest is trained on a subsection of the total available data and then combined using a weighting approach rewarding generalizability within the training set.

Random Forest is itself a highly popular ensemble machine learning algorithm designed to reduce overfitting and handle high-dimensional data, and has shown immense success in many scientific realms. The method was introduced by Leo Breiman in 2001 and has since been adapted to handle a wide variety of data types and applications, including missing data imputation, survival analysis, and unsupervised clustering [5] [6]. Ramchandran et. al. (2020) previously showed that in the presence of heterogeneity in both the outcome model and distribution of the features across multiple training studies, building ensembles of forests each trained on a single study produced significantly more accurate predictions than training a single forest on all of the available data with an equivalent number of trees [7]. Additionally, these results held when heterogeneity in the covariate-outcome relationship was removed; when trained on randomly-chosen subsections of a large single dataset, forest-based ensembles considerably improved upon a single forest trained on the entire dataset. We thus hypothesize that for datasets containing clusters, data can be even more optimally partitioned using clustering algorithms in order to form ensembles of forests that outperform the classic Random Forest algorithm.

The general strategy of utilizing clustering algorithms to partition data and build ensembles of learners trained on each discovered cluster has been previously explored in Deodhar and Ghosh (2007) [8]. They primarily considered logistic and linear regression as base learners, in which the full ensemble itself represents a soft addition of random effects to the chosen learning algorithm. Similarly, ensembles of Random Forest learners can be themselves thought of as modified Random Forests, with the primary differences being in how the data is used to train each component tree as well as how the trees are weighted within the overall ensemble. The clustering step efficaciously removes within-cluster feature distribution heterogeneity while maximizing cross-cluster heterogeneity, and thus the strategy of ensembling learners trained on these disparate partitions reflects the common ensemble learning standard of combining dissimilar learners in order to improve prediction generalizability [9] [10]. In fact, this is the very paradigm that motivates the classic Random Forest algorithm, in which the randomness of the bootstrapping step and the subset of total variables available at each split serve to create diversity in the trees that comprise the ensemble and thus lower its variance [11] [12].

We propose a strategy of training ensembles of forests in which clusters of points that produce highly generalizable predictors are given greater influence. In this paper, we introduce the **Cross-Cluster Weighted Forest**, a learning approach with the following steps: first, the dataset is separated into clusters using k-means or other clustering algorithms, with Random Forests subsequently trained on each cluster. Finally, stacked regression weights are employed to construct the overall ensemble, in order to up-weight individual predictors that show stability across clusters. We illustrate that our method greatly improves upon the existing paradigm across a variety of realistic scenarios as well as real biological data naturally divisible into clusters. Our overall aim is to improve upon the traditional Random

Forest algorithm in the presence of heterogeneity in the distribution of the features, in order to form more replicable predictors.

We begin this article by describing the Cross-Cluster Weighted Forest approach and the additional ensembling methods we will be comparing throughout our analysis. We then present the results of our numeric explorations on its performance on single datasets containing clusters, organizing our discussion by isolating the beneficial effects of each step in our algorithm on the behavior of the overall ensemble. We additionally determine the robustness of our method to realistic variety in data generation, demonstrating that our proposed approach has greater generalizability across scenarios to all other methods considered. Next, we explore the multi-study setting, and illustrate that the addition of the clustering step within our framework in fact improves upon the existing paradigm of simply ensembling forests trained on each study. Finally, we consider real gene expression and clinical data plausibly containing heterogeneity in the distribution of the features, and demonstrate our method's superior performance over classic Random Forest.

2 Methods

The Cross-Cluster Weighted Forest (CCWF) approach is outlined in Algorithm 1. We denote by $\hat{Y}_j(\mathbf{x}_\star)$ the prediction of the forest trained on cluster j on new point $\mathbf{x}_\star$ for $j = 1, \ldots, k$, and by $\hat{Y}_E(\mathbf{x}_\star)$ the prediction of the overall CCWF on $\mathbf{x}_\star$. Classically, the Random Forest

Algorithm 1 Cross-Cluster Weighed Forest (CCWF)

Set the number of clusters $k \geq 2$; then,

1. Run a clustering algorithm on the entire dataset
2. For$j = 1, ..., k$
   - Train a Random Forest with 100 trees on the jth cluster
   - Compute $\hat{Y}_j(\mathbf{x}_\star)$
3. Compute stacked regression weights for all cluster-level forests, $w_1, ..., w_k$
4. Construct the ensemble-level prediction: $\hat{Y}_E(\mathbf{x}_\star) = \sum_{j=1}^{k} w_j \hat{Y}_j(\mathbf{x}_\star)$

algorithm trains trees on bootstrap samples of the full dataset and weights tree-level predictions equally in the final ensemble. Utilizing bootstrap samples of the same size as the original dataset has never been proven to be optimal - it is simply an approach that works well heuristically. The introduction of a non-random data partitioning step in the training of tree-based ensembles has the potential to produce further improvements. Throughout our simulations, we utilize k-means clustering as our choice of clustering algorithm in step (2), both for its advantageous and well-characterized properties such as convergence, ability to scale, and generalization to clusters of different shapes and sizes, as well as for its relative simplicity and ease of interpretation. However, any clustering method may be substituted into the general framework.

The majority of previous work connecting Random Forest with clustering utilizes Random Forest itself as a clustering method [13] [14] [15]. To our knowledge, the only manuscript which displays conceptual similarity in broadly using the data partitions produced by clustering algorithms to train ensemble learners including Random Forest is Trivedi et. al. (2015) [16]. However, they present a different paradigm, in which the prediction for a given test point involves first running k-means for several different values of k, approximating the cluster from each iteration that most closely fits the test point, and then ensembling learners trained only on these successive clusters in order to form the final prediction. This requires assigning the test point to a cluster for each value of k, which may or may not be an optimal characterization of its distribution, and then removing all other data points from consideration. Their results showed marginal improvements in certain scenarios of their ensembling approach using Random Forest over the use of a single forest, with only a subsection of those showing statistical significance.

To construct the ensembling weights in step (3) of our algorithm, we adapted the multi-study version of the stacked regression method, an approach which rewards proven cross-cluster generalizability of cluster-specific learners [4] [7]. Stacked regression forms linear combinations of multiple predictors that has been shown to improve upon the performance of any single component [17]. The weights given to each predictor are determined by cross-validation and least squares regression with L_2-norm regularization. Our extension of stacking to the multi-cluster context proceeds as follows: once the training set is split up into k subsections and forests are trained on each partition, all forest-level predictions are then 'stacked' into matrix $\mathbf{T} = [\hat{Y}_1', ..., \hat{Y}_K']'$, where $\hat{Y}_j = [\hat{Y}_{1j}...\hat{Y}_{kj}]'$ for $j = 1, \ldots, k$; $\hat{Y}_{ij}$ is the vector of predictions of forest j on cluster i, and $\hat{Y}_j$ is the stacked vector of all predictions made by forest j on every cluster within the training set. $\mathbf{T}$ is $n \times k$, with n equaling the total number of observations in all datasets. The true outcomes are similarly aggregated across the entire training dataset into the $n \times 1$ vector Y. Y is then regressed against T with a ridge penalization and non-negativity constraint. The learner-specific weights $\mathbf{w}_{stack}$ are determined by solving $\min_{\mathbf{w}_{stack}} ||(Y - (\mathbf{T} \times \mathbf{w}_{stack}))^2||$ such that $\mathbf{w}_{stack} > 0$ and $||\mathbf{w}_{stack}^2|| \leq \lambda$, where λ is optimized using the cross-validation procedure in the `glmnet` package in R [18].

In the following section, we assess the utility of the CCWF algorithm across various scenarios of naturally clustered data which will be described in more detail in Section 3. Throughout, the abbreviation *Cluster* will refer to training an ensemble through the CCWF approach. We additionally compare the performance of CCWF with other forest-based ensembles representing variations from Algorithm 1, in order to isolate the effect of each step on conferring advantages to the method. The moniker *Random* will indicate the ensemble formed by using k equally sized random subsections of the dataset instead of clusters. *Multi* will refer to training a forest on each of the true clusters (which are known from the simulation) and combining again using stacking weights to form the final ensemble. The name derives from its similarity to the multi-study paradigm, in which learners are trained on known studies prior to ensembling. Finally, the term *Merged* will refer to a single forest with $k \times 100$ trees trained on the entire dataset. We kept the number of trees in the overall ensemble consistent across methods, so that the comparison consists solely in how the training data is differently used to train the trees in each ensemble. We set the number of trees per forest at 100 (other than the *Merged*, as previously described) based on the observation that higher number of trees did not significantly improve the prediction results nor alter the

relationship between approaches.

3 Numeric Explorations: Strategies and Results

In this section, we use simulations to evaluate the overall behavior of the ensembling approaches described above on a variety of data generating mechanisms and outcome-covariate models that we would expect to find in real genomic datasets. We begin by delineating our choice of data generation and simulation strategy. We then discuss the importance of data-subsetting prior to the training of each component forest, where we particularly focus on the benefits of using clustering algorithms as opposed to other methods. We then examine the robustness of CCWF to realistic data situations, in which we vary parameters such as the signal to noise ratio, the number of true clusters within the dataset, and the total sample size. Finally, we determine the effect of ensemble weighting strategy on overall performance, specifically highlighting the advantages of stacked regression weights over simple averaging as well as how the differences in the distribution of weights across the ensembling methods correlate with prediction accuracy. Throughout this section we report averages over 250 simulations per scenario, with 95% confidence bands computed as mean $\pm 1.96\times$ standard error.

3.1 Generating multivariate clustered data

We utilized two primary approaches to generate the necessary clustered datasets for simulations. The first was to draw clusters from multivariate gaussian mixture models, in order to easily simulate many datasets with analytically tractable characteristics. We used the implementation within the R package `clusterGeneration`, with specific parameters outlined in the following section [19]. However, a major disadvantage is that simulated clusters using this more simple setup may lack important features of the real biologic clusters we would want the methods to work well on in practice.

In order to address this potential pitfall, we additionally considered simulating clustered datasets that more closely mimic what we find in nature. Specifically, we utilized the method proposed by Waller et. al. and implemented in R via the 'monte' function within the `fungible` package [20]. Their algorithm produces artificial plasmodes with clusters that resemble authentic clusters on important features such as cluster size, shape, and orientation. Throughout, we evaluated our ensembling approaches on clusters generating using both methods for a variety of different scenarios, as described below.

3.2 Simulation setup

For N = 250 iterations at each set of generative parameters, we simulated a training dataset with *ntrain* number of clusters and *ntest* test datasets with two clusters using either the mixture model or the plasmode framework, all with *ncoef* number of covariates. At baseline, $ntrain = ntest = 5$ and $ncoef = 20$. Per iteration, we then randomly sampled 10 of these covariates to create a linear data-generating model with gaussian noise, with coefficients drawn uniformly from $[-5, -0.5] \cup [0.5, 5]$ so that each has a non-zero contribution to the outcome.

We additionally considered non-linear relationships of the covariates to the outcome, including the addition of quadratic terms, between-covariate interaction terms, and binarization of the covariates before application of the linear model in order to form a step function. Since the focus of our investigation is to evaluate the effect of feature distribution heterogeneity on ensemble performance, we kept the covariate-outcome relationship relatively similar across clusters, with the amount of coefficient perturbation between clusters uniformly drawn from the interval [0, .25]. The size of each cluster was set at baseline to be 500, resulting in training datasets with 2500 total samples. Using either the `clusterGeneration` or `fungible` packages, the level of between-cluster separation was set at median values, resulting in clusters with some overlap that would still ostensibly be distinguishable using clustering strategies. All modifications to this baseline strategy are outlined in the relevant results sections.

3.3 The importance of data subsetting on prediction accuracy

We commence with an exploration of how the choice of data partitioning method affects the accuracy of the resulting ensemble. The *Cluster* and *Random* methods train the component forests on the same number of subsections - however, the composition of each partition is the primary differentiation between the two. The *Multi* uses a fixed number of subsections corresponding to the true clusters, while the *Merged* simply uses the bootstrap sampling inherent to the Random Forest algorithm to build each tree. We denote by k the number of subsections used to construct the *Cluster* and *Random*, and investigate throughout this section the effect of increasing k on both approaches. We additionally examine whether changes in the distribution of the training clusters influences the accuracy and relationship between all four methods considered.

Figure 1 displays the result of varying both the value of k and the data-generating mechanism for a total of 6 different scenarios. Overall, we observe that the *Cluster* and *Random* improve upon ensembling using the true clusters or the entire dataset for values of k higher than the true number of clusters. The level of improvement that the k-dependent approaches show over the *Merged* (and *Multi*) is substantial, typically over 20-30% at optimal values of k. On the other hand, the *Multi* is inconsistent in its relationship to the *Merged* over all scenarios considered, showing that the perhaps more intuitive approach of splitting the data into its true clusters does not necessarily produce the best results. The *Cluster* and *Random* are robust to changes in the outcome model and the distribution of the data, as evidenced by the similar performance patterns seen over three separate covariate-outcome relationships and two covariate-generating mechanisms. Finally, in the majority of cases, the *Cluster* exhibits superior accuracy over the *Random*, indicating that using k-means to create the partitions is generally preferable over randomly sub-setting. As k increases, the accuracy of both the *Cluster* and *Random* proportionally grow until finally plateauing, typically at the point in which each forest is being trained on 35-40 observations out of 2500 total. The average depth of the trees within each forest decreases correspondingly with the sample size per partition. Tree depth is a measure of model complexity, indicating that for high values of k, we are essentially ensembling locally weak learners in an approach akin to local smoothing.

The differences between the *Cluster* and *Random* ensembles reveal the particular advantages of using clustering as the partitioning strategy. The data sub-setting step primarily serves to reduce the range of the covariate space used to train each tree, and we see through-

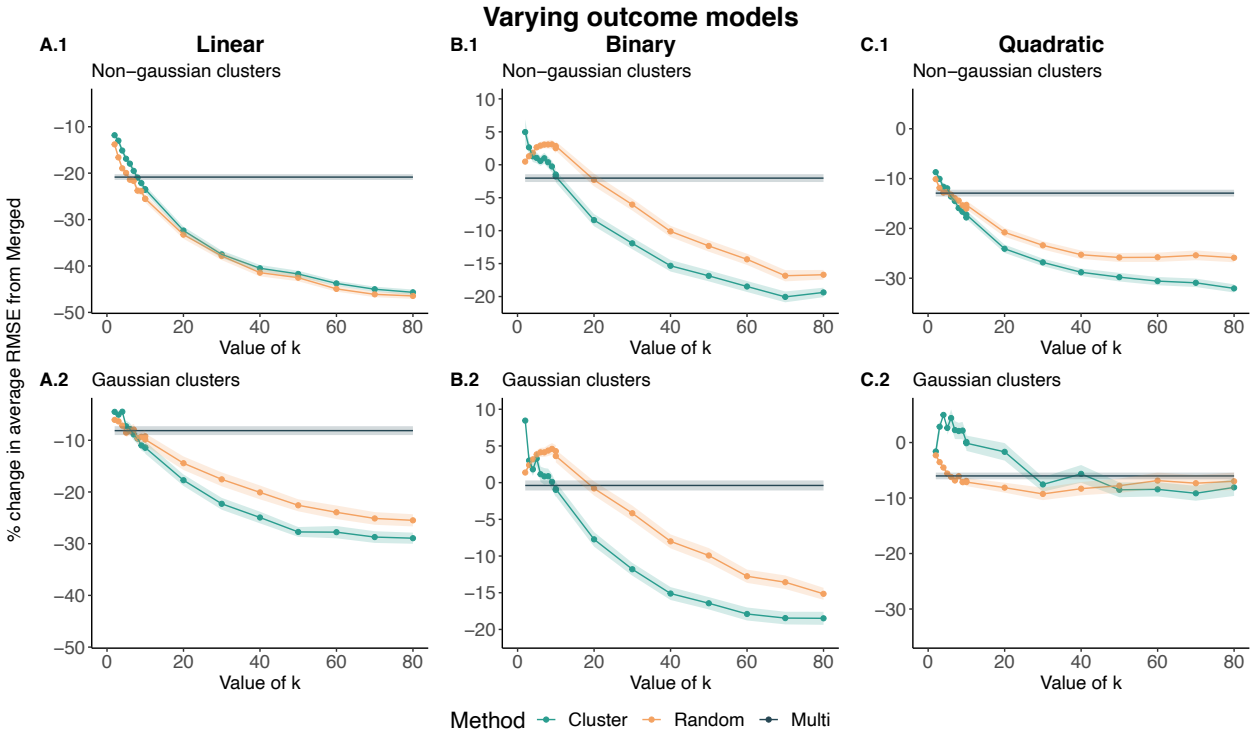

Figure 1: Percent change in average RMSE of ensembling approaches (color labeled) compared to the Merged across different data-generating scenarios, as a function of k. The first row depicts results using the non-gaussian cluster simulation approach, while the second row uses a gaussian framework. **(A.1-A.2)** A linear model was used to generate the outcome from the covariates. **(B.1-B.2)** The binary outcome was created by using a cutoff from the linear model to create a binary step function. **(C.1 - C.2)** Quadratic terms for two of the variables were added to the linear outcome-generating model.

out Figure 1 that in general, increasing the number of subsections increases the performance of the ensemble. This additionally supports the observation that for k less than the number of true clusters, the *Multi* approach typically outperforms the *Random* and *Cluster*. The manner and efficacy to which this partitioning step is performed correlates with the prediction performance. In fact, when comparing the association of prediction accuracy to the degree of range reduction of the covariate space, we find a clear relationship; at the optimal value of k, randomly sub-setting results in a slightly higher average covariate range than k-means (**4.22** CI:(4.13, 4.31) and **3.17** CI:(3.08, 3.26) respectively), while the true clusters have a significantly larger average range (**7.33** CI:(7.21, 7.45)). Furthermore, using k-means (or other clustering methods) results in not only greater range reduction but also greater between-cluster separation than randomly sub-setting, which additionally contributes to the better performance of the *Cluster* over the *Random*. Clustering allows the learners to be as uncorrelated and dissimilar as possible, a framework that has been previously shown to be effective in other boosting and sequential ensembling approaches [21] [9].

It is additionally important to note that projecting the distinct partitions of the total joint covariate space produced by clustering onto the 1-dimensional space corresponding to a single variable results in ranges that are highly overlapping between clusters. This mitigates

perhaps the greatest downside to the clustering strategy of minimizing the euclidean distance, as Random Forest lacks the ability to extrapolate beyond the data it is trained on. However, the marginal degree of range restriction for a single variable is highly variable across clusters, allowing in aggregate over all cluster-level forests for total coverage of not only the complete range but also the fluctuating relationship between covariates in different regions of the joint distribution. While this presents a slight trade-off in sacrificing accuracy in the areas of highest density for better precision at the tails, the loss is not significant enough to counteract the benefits.

We next examine the implications of these improvements on our understanding of the classic Random Forest algorithm. Each tree within a Random Forest is trained by formulating successive partitions of the bootstrapped covariate space that maximize the amount of information gained by each split. A prediction for a new point is then obtained by averaging the outcomes of the training data in the same leaf node as the test point across all trees. In this way, the Random Forest algorithm can itself be seen as an combination of learners across different subsections of the data - however, since the available training data for each learner is simply obtained by bootstrapping, the partitions influencing final predictions are not achieved by optimization of any criterion, simply randomness.

While this is often a very powerful paradigm, the improvement of the *Cluster* over the *Merged* indicates that in the presence of heterogeneity in the training data, more optimal data partitions for the training of each component tree in the overall ensemble can be obtained. Clustering the data prior to training restricts each forest to a smaller subset of the covariate space. The result is that these forests can more effectively form splits to learn nuances in this restricted space that would not be possible were the same number of total trees trained on the entire dataset. Insight into this phenomenon may be gleaned by considering how the bootstrap samples of the data used to train each tree are differentially obtained between the two approaches. A forest trained on a partition mostly comprising of observations from one of the tails of the joint distribution of the covariates will consist of trees trained on bootstrap samples of only these observations, resulting in a far greater representation of these observations within trees in the overall ensemble than in trees trained on bootstrap samples of the merged dataset. Supplementary Figure S1 displays the differences in how the *Cluster* and *Merged* learn the underlying covariate-outcome function when there is only one covariate. The *Cluster* more effectively learns the function at the tails of the covariate distribution whereas the *Merged* primarily learns the highest density regions.

Interestingly, the clusters produced by k-means do not correspond with the true clusters for either simulation strategy, or when tested on real genomic clustered data. This holds even when the value of k is equal to the number of true clusters. Furthermore, across all values of k considered, each k-means-induced cluster typically ($> 90\%$ of time) contains observations from all of the true clusters - as k increases, the total number of observations from each true cluster simply decreases, but there is still full representation across clusters. This further supports that the true clusters are not necessarily the best partitions of the data for building ensembles of forests, and that in fact the k-means algorithmic strategy of producing clusters with minimal variance within and maximal variance across is more effective at capturing feature distribution heterogeneity regardless of underlying data structure.

Silhouette analysis is often utilized to choose the optimal value of k for k-means; the k that maximizes the silhouette score is the one for which the ratio of between-cluster variance

to within-cluster variance is maximized. We found that the *Cluster* strategy was only more effective than the *Merged* if silhouette analysis indicated that the optimal k was greater than 1, regardless of whether the dataset was actually constructed to have clusters or not. Furthermore, even when varying the separation of the simulated clusters from mostly overlapping to completely separated, we found no exceptions to this observation. Of course, we do remark that in the case of having completely separated true clusters, it is unlikely that silhouette analysis will indicate the optimal number of partitions to be equal to 1, but regardless, the algorithmic clusterability of the dataset is the criteria that matters in determining the efficacy of the *Cluster* approach, not necessarily the presence of true clusters. We found similar results when applying other clustering methods instead of k-means; in general, the unsupervised clustering step seems to provide a good metric of whether the supervised ensembling step will perform well.

3.4 Robustness of method to realistic dataset variety

Our next simulations evaluate the robustness of these methods to a range of dataset characteristics likely to be encountered in real data, while holding the value of k constant at the optimal value chosen by Silhouette analysis. We determine how prediction performance is affected by the number of true clusters in training, the sample size of the training set, and the magnitude of the relationship between the covariates and the outcome. For this set of simulations, we are only showing results using a quadratic outcome model and the non-gaussian cluster generating setup, in order to present a fairly complex and realistic data paradigm.

In the simulation used to generate Figure 2a, the norm of the coefficients relating the covariates to the outcome in the quadratic model was varied in order to determine how well the approaches picked up differing signals from the covariates. We see that as the signal from the covariates increases, the magnitude of the change in performance from all ensembling methods over the *Merged* expands significantly, eventually plateauing at rates of improvement ranging from 20-60%. The *Cluster* is by far the most effective learner, especially at higher coefficient norms, while the *Random* displays a similar trajectory of improvement, albeit significantly lower than that of the *Cluster*. Furthermore, the vastly decreased improvement of the *Multi* over the *Merged* indicates that increasing the signal does not necessarily signify that the true clusters will more effectively pick it up. This evidences the supposition that reducing the amount of within-cluster variation (as most effectively accomplished by clustering for optimally high values of k) allows for more accurate discernment of the covariate-outcome relationship, since it is no longer being obscured by heterogeneity in the distribution of the covariates.

Next, we evaluated the effect of varying the number of true clusters within each training dataset while keeping the total sample size constant at 2500. Figure 2b demonstrates that as the number of true clusters increases, the *Cluster* method remains significantly better than all other approaches, while the performance of the *Multi* catches up to that of the *Random* at higher numbers of true clusters. This latter phenomenon makes sense in light of what we have seen before, as the number of true clusters determines the number of forests in the *Multi*. As illustrated in Figure 1, increasing the amount of ensemble members through k augments prediction accuracy, whereas the *Multi* was set in those simulations to always include only 5 component forests. However, when the number of true clusters is high, the difference between

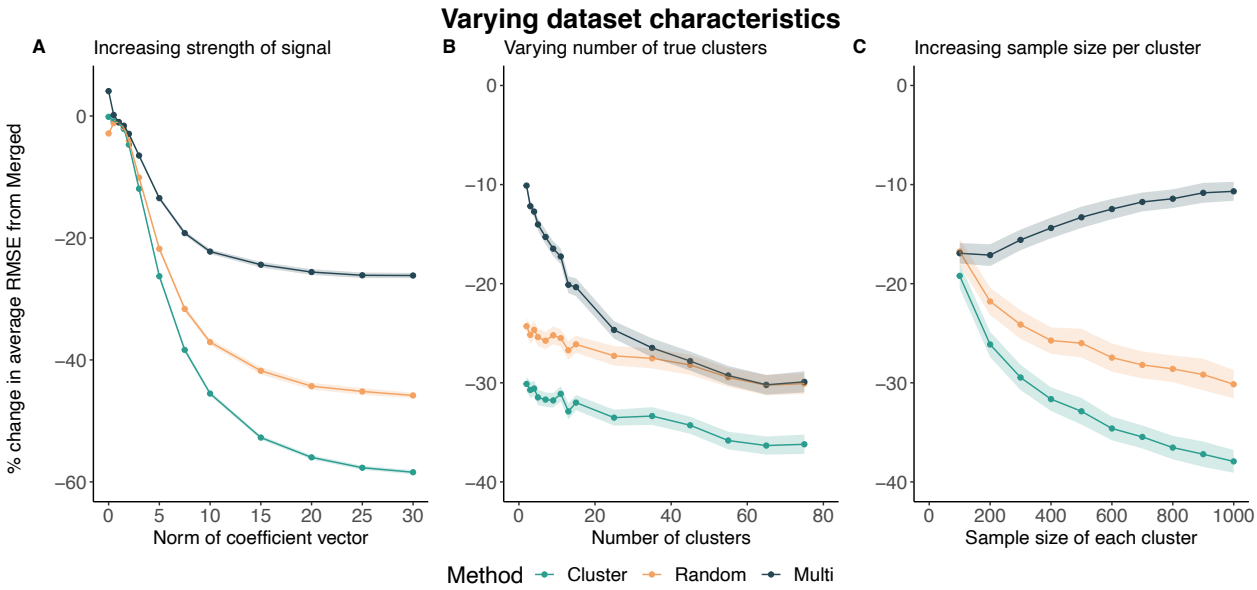

Figure 2: Percent change in average RMSE of ensembling approaches (color labeled) compared to the Merged across different data-generating scenarios. All simulations used a quadratic outcome and the non-gaussian cluster generating algorithm. **(a)** Increasing the magnitude of the coefficients in the outcome-generating model to determine the effect of the signal from the coefficients on prediction accuracy. **b** Varying the number of true clusters within the training set, while keeping the total sample size constant at 2500. **(c)** Increasing the sample size per cluster, while keeping the total number of clusters per dataset constant at 5.

the *Cluster* and the other two ensembles remains constant, highlighting that the benefits of clustering persist no matter the true composition of the data.

In the simulation used to generate Figure 2c, we set the number of clusters per training set constant at 5 while increasing the sample size of each dataset incrementally from 500-5000. We observe that as the sample size of each cluster grows, the difference in prediction accuracy between the *Cluster* and the other approaches similarly increases. On the other hand, the *Multi* becomes less distinguishable from the *Merged*, primarily because the accuracy of the *Merged* increases proportionally to the sample size. However, the *Cluster* and *Random* methods continue to improve upon the *Merged* regardless of the sample size, since the optimal number of partitions k increases proportionally with the sample size.

3.5 Importance of ensemble weighting strategy on performance

Our next simulations explore the importance of the ensemble weighting strategy on prediction performance. We specifically highlight the importance of using stacked regression weights over equally weighting component learners or other weighting approaches.

The weighting step is critical in determining the success of the ensemble; for instance, if we weight the cluster-trained forests either equally or by the inverse of their sample sizes, the resulting ensembles often perform as badly as the *Merged*, or worse. Table 1 displays the

Value of k	2	5	10	20	30	50	70	80
Simple Averaging	11.93	29.97	43.60	59.63	68.17	79.09	86.65	89.48
Stacked Regression	-9.88	-13.35	-19.73	-25.21	-27.73	-29.99	-31.96	-32.27

Table 1: Simple averaging vs. stacked regression weights. Percent change in average RMSE from Merged for differently-weighted ensembles built on clusters determined by k-means, using the non-gaussian dataset simulation framework. The first row shows results for weighting each forest equally and the second row depicts results from using stacked regression weights.

percent change in average RMSE from the *Merged* achieved by ensembles of forests trained on clusters determined by k-means and then combined using either simple averaging or stacked regression weights. We observe that as k increases to its optimally performing value of around 80, the stacking approach continuously advances while equal weighting progressively declines. In fact, the best number of subsections for equal weighting is 2, at which point it is still significantly worse than merging. This phenomenon indicates that clearly not all subsets from k-means produce accurate predictors, but certain subsets in combination produce better predictions than the *Merged*. Stacking up-weights the subsets that show the best cross-cluster prediction ability, which likely translates to accuracy and robustness to test data. If a cluster-level forest is performing well on other clusters, this indicates that it has more effectively learned the true covariate-outcome rule and is generalizable to training data dissimilar from its own, as clusters are designed to have as much distributional separation as possible.

Figure 3 displays the distribution of stacking weights across ensembling methods using the gaussian cluster-generating framework and a linear outcome model, although results for the non-gaussian simulations were virtually indistinguishable. In looking at the distribution for the highest-weighted forest per ensemble, it is clear that the *Cluster* method most strongly up-weights beneficial predictors out of all approaches considered, demonstrating the presence of certain clusters that produce highly generalizable forests. Furthermore, we see significantly more upweighting in the *Cluster* approach as compared to the *Random*, whose stacking weights all occupy a more restricted, and lower, range. All forests within the *Random* approach are more similarly performing, which stands to reason given that all the subsections used to train each forest are randomly selected. The benefit of clustering thus lies in producing more optimal subsets for training, although we note through both the results from equally weighting all predictors as well as the distributional range of stacking weights that only some of these clusters produce predictors that are decidedly effective. To the latter point, the *Cluster* produces a larger difference between the most up-weighted forests and the weights given to the rest of the ensemble members than the *Random*. The *Multi* displays the smallest difference of the three, which corresponds with its diminished performance.

Stacking can considered as an indirect form of reweighting the influence of each training data point on the overall predictor. The clustering step helps the stacking algorithm to identify the most effective points to up-weight by grouping together data that are likely to exert a similar influence. Points belonging to clusters that are not weighted as highly are still necessary for the overall ensemble function, and should not be removed entirely - when we regularized the stacked regression weights using Lasso instead of Ridge, the effect was to zero

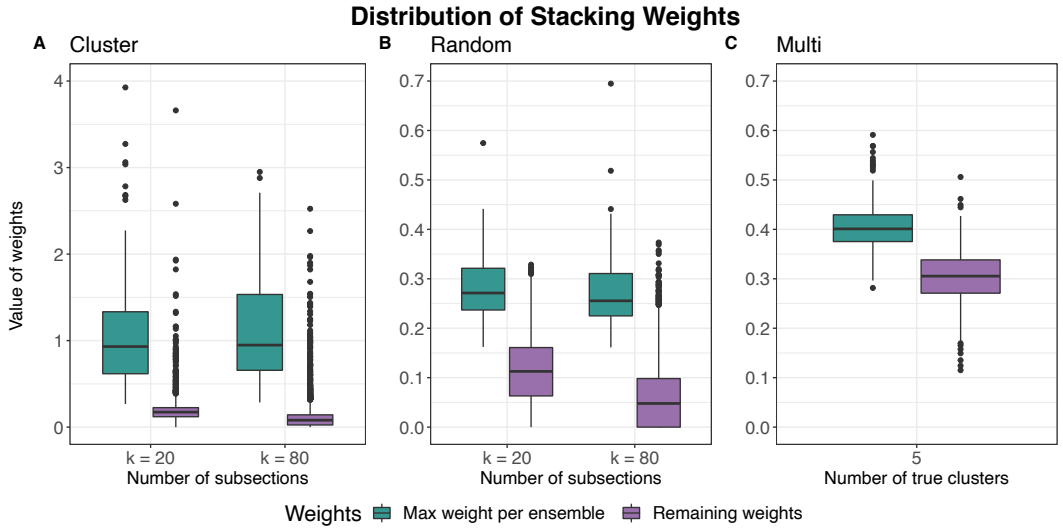

Figure 3: Distribution of the ensemble weights determined by stacked regression for **(a)** the *Cluster*, **(b)** the *Random*, and ((c)) the *Multi* for $k = 20$ and $k = 80$ for the first two methods, and 5 true clusters for the latter. We used the gaussian cluster-generation framework. The distribution of the largest weight per ensemble is depicted in green, while the rest of the weights are visualized in purple. Results are shown over 100 iterations at each value of k.

out the contributions of several clusters per ensemble. The resulting predictors were far more variable in their accuracy across data-generating scenarios, likely because they adapted too strongly to random noise in the training data. Thus, stacking with a Ridge constraint gives more importance to data points in heavily up-weighted clusters while still recognizing the necessity of the the remaining clusters to prevent overfitting and promote generalizability.

3.6 Extension to the Multi-Study setting

We next examine whether we can more optimally partition the total amount of data when multiple studies are available for training that measure the same covariates and outcome variable. The traditional multi-study ensembling paradigm is to train a single learner on each study and combine using some weighting strategy, such as stacking [4]. This is analogous to the *Multi* method when we are able to separate data into its true clusters. We now explore whether training k-means on the the merged data (comprising covariate data from all of the training studies) produces improvements comparatively to the single dataset setting. We furthermore evaluate the performance of these approaches on real gene expression data from the CuratedOvarianData repository from Bioconductor in R [22]. Finally, we explore whether the general strategy of ensembling learners built on clusters also works for Neural Nets, and compare the results to those when using Random Forest.

The general framework for the simulation setup used to create Figure 4 was drawn from Ramchandran et. al. (2020) [7] [4]. CuratedOvarianData provides data for gene expression meta-analysis of patients with ovarian cancer. In this study, we used all 15 studies in CuratedOvarianData that include survival information without any missing data in the features. For N = 250 iterations per value of k, we randomly separated the 15 datasets from Curate-

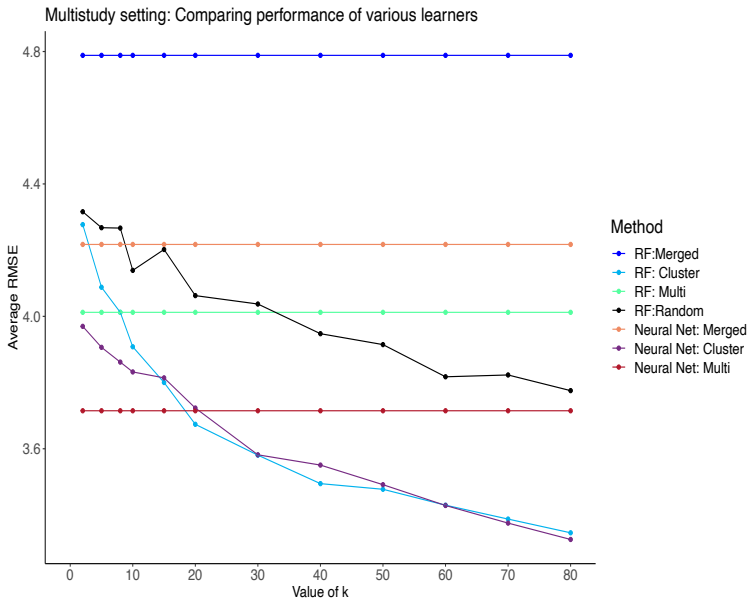

Figure 4: Average RMSE's of ensembling approaches (color labeled) as a function of k. 20 total covariates, 10 associated with the outcome; nonlinear outcome model of the form $y = \beta^T \mathbf{X} + 4.4x_1 - 1.8x_2 + 10\sin(10\pi x_1)$.

dOvarianData into 10 training and 5 validation sets. We then generated the outcome using a complex non-linear model in order to test the ability of the candidate learning methods to detect more difficult covariate-outcome relationships. We simulated baseline levels of coefficient perturbation per study as described in the Methods section. Using either Random Forest or Neural Nets as the base learner, we then constructed ensembles using the four main approaches compared throughout this paper: the *Merged*, *Cluster*, *Random*, and *Multi*. The *Multi* in this case trained a learner on each study to form the final ensemble. All ensembles were built using stacked regression weights with a ridge constraint.

We commence by discussing the performance of the Random Forest-based ensembles. In the multi-study setting, ensembling based on k-means clustering is significantly more accurate than ensembling based on study membership for values of k higher than the number of training studies. The relationship between approaches follows exactly the pattern we observe in the single clustered dataset setting, with the *Cluster* performing the most effectively, the *Random* performing on a similar trajectory but less accurately, the *Multi* at an even lower level, and the *Merged* by far the worst of the four. The true cluster or study-membership does not represent the most effective data partitioning for Random Forest; again, it is more effective to subsection the data based on minimizing within-cluster heterogeneity and maximizing across. Overall, these results suggest that for either single or multiple training datasets, the *Cluster* approach should be used for Random Forest learners.

We next investigate the use of Neural Nets in this setting and compare the results with Random forest. Figure 4 displays that the *Cluster* method with Neural Net component learners improves upon training each learner on the true studies or training a single Neural Net on the merged data. The level of improvement of the *Cluster* over the *Merged* is less than that for Random Forest, as the Neural Net *Merged* learner is significantly more accurate

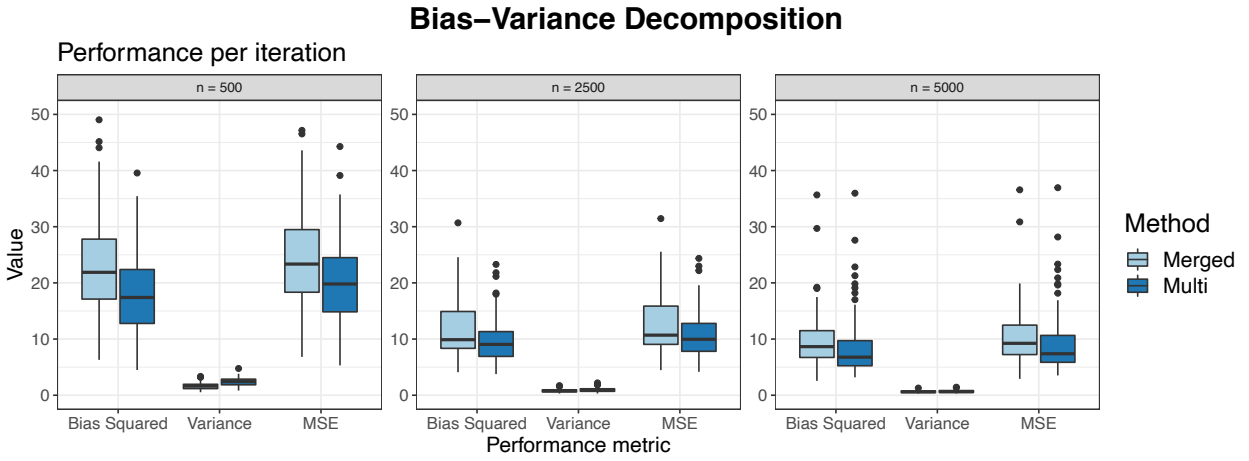

Figure 5: Squared bias, variance, and MSE of the Merged and Multi learners (color coded) for datasets ranging from 500-5000 total samples (100-1000 samples per cluster). Datasets were generated using the gaussian cluster framework. Each panel shows the breakdown for a given sample size.

than its Random Forest counterpart. Interestingly, the *Cluster* method produces almost identically accurate predictors using either algorithm its base learner. While the traditional multi-study paradigm or simply training a single learner on the merged data favors Neural Nets, Random Forest catches up when trained on clusters. This displays the magnitude of improvement capable by the cluster-based ensemble framework when applied to Random Forest, and further highlights that even when the traditional Random Forest algorithm falls short of the performance of a Neural Network, the *Cluster* approach can elevate the two to the same level. In fact, the *Cluster* strategy can itself be compared to the structure of a convolutional neural network, in which the clusters represent the convolutional layers and the stacking weights delineate the relationship between each later. While this analogy does not numerically explain the similar performance of the two algorithms, it provides intuitive insight into why we may be seeing these results.

3.7 Examining the bias and variance

Our final set of simulations decomposes the MSE of the *Merged* and the *Multi* into its bias and variance terms in order to individually characterize each. As the sample size of the training dataset increases, the MSE of both ensembling approaches decreases as expected. However, there is a far more minimal decrease from $n = 2500$ to 5000 than from 500 to 2500, indicating that there is an asymptotic limit to how well the ensembles can perform. Most strikingly, across all sample sizes considered, the squared bias term comprises almost the entirety of the MSE while the variance contribution is minimal. All improvements in performance from the *Multi* over the *Merged* arise through decreasing the bias of the ensemble - in fact, the variance slightly increases between the two but is proportionally so small as to have almost no effect. We limit our comparison to the *Multi*, since this ensemble construction method can be analytically characterized much more easily than the *Cluster* or *Random*. These results indicate that the squared bias defines the overall performance of the *Multi* ensemble, and

that any theoretical analysis of this general approach can be largely restricted to the bias term.

4 Data Application

To explore the performance of these classifiers on real biological data with natural feature-outcome relationships, we considered datasets measuring gene expression and clinical data which plausibly contain clusters.

4.1 Low Grade Gliomas (LGG) Dataset

We applied our methods to the Low Grade Gliomas (LGG) dataset within TCGA [23]. We considered one binary and one continuous outcome in order to assess the performance of the ensembles on both regression and classification tasks. The binary variable measures the tumor grade, in which the value 0 indicates the presence of a low-grade tumor (WHO grades II and III) and the value 1 indicates full-blown glioblastoma (WHO grade IV). The continuous outcome measures the number of mutations present in each sample. These two outcomes were chosen for their importance in diagnosing and informing the treatment of glioblastoma. We additionally experimented with two different sets of covariate data. The first represents molecular profiling data (for example, measures such as percent aneuploidy and TERT expression), in which there are 513 total patients and 50 covariates. Twelve of the covariates are continuous while the remainder are categorical; missing covariate data was imputed using Random Forest through the mice package in R prior to training. We additionally considered gene expression data for the same set of patients; the original authors previously demonstrated that the patient gene expression data is separable into clusters using unsupervised approaches, so we included this dataset in our analysis to determine the efficacy of the clusters they discovered within our ensembling paradigm [23].

We used a variable selection and clustering strategy based on `mclust` (as implemented in the `vscc` package in R) for the molecular profiling covariate data [24] [25]. We chose this method over k-means due to its simultaneous ability to perform variable selection and cluster the data without prior specification of the number of clusters k, and to additionally display that the general cluster-weighted ensemble framework is robust to the choice of clustering method. `mclust` performs model-based clustering using Gaussian mixture models, in which the variable selection step chooses the candidate variables that result in clusters with the highest likelihood. The algorithm can only handle uniform data types, so we chose to solely restrict the clustering and variable selection steps to the 12 continuous covariates. No further clustering was necessary for the gene expression data beyond the analysis previously conducted by the original authors. Per iteration, we randomly chose 100 sample points for the test set and used the rest of the samples for training, ultimately repeating this procedure 500 times to obtain the distributional and median measures of prediction accuracy.

For this analysis, we consider modified definitions for the the ensembling strategies we have investigated thus far. The *Merged* will have the same definition as before, representing a single forest with 500 trees trained on the entire dataset with all 50 covariates. We denote by *Subset Merged* a single forest with 500 trees trained only on the 12 continuous candidate variables

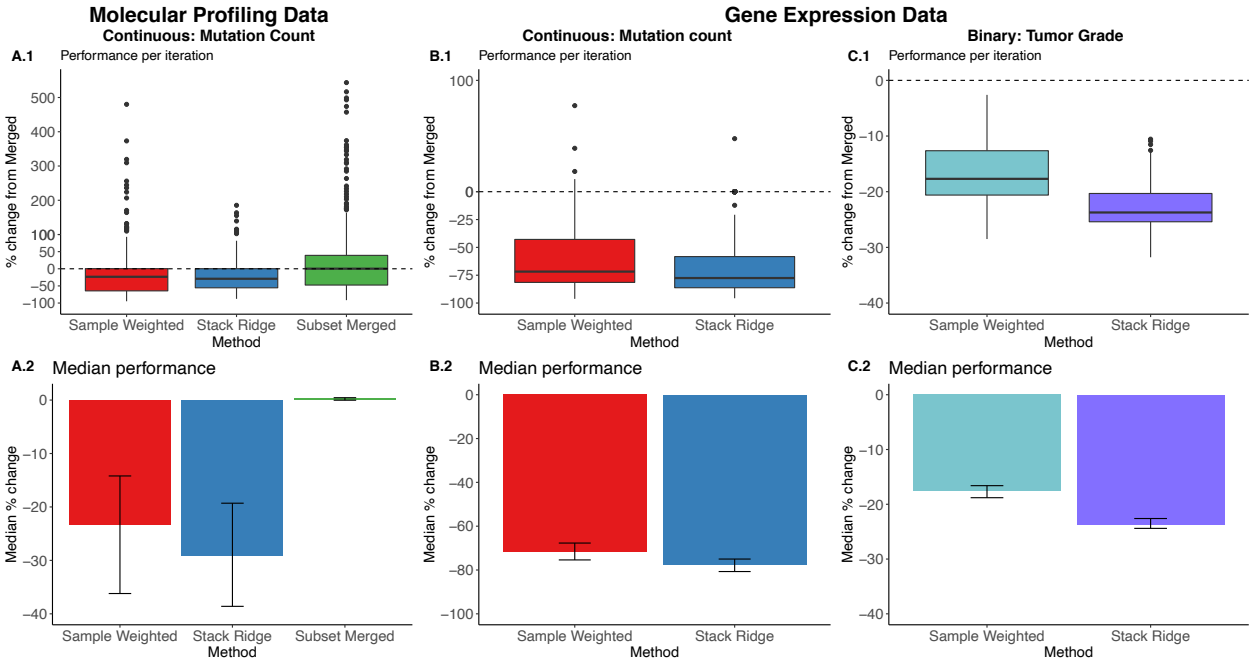

Figure 6: Percent change in average RMSE of ensembling approaches (color labeled) compared to the Merged across two sets of covariate data and two different outcomes from the LGG study. The top row shows 500 iterations of splitting the available data into training and test sets, while the second row displays the median over all iterations with associated confidence intervals. **(a)** Molecular profiling covariate data, mutation count outcome. **(b.1)** Gene expression covariate data, mutation count outcome. **(b.2)** Gene expression covariate data, tumor grade outcome.

used to cluster the molecular profiling data. *Sample Weighted* indicates first clustering the training data using the `vscc` package, training a forest with 100 trees on each cluster, and ensembling using weights proportional to the sample size of each cluster. While clusters are determined by only considering the continuous covariates, all 50 covariates are utilized during training. Finally, *Stack Ridge* represents the clustering and ensembling method described above, instead employing stacked regression weights with a ridge constraint to form the final predictor.

4.2 Results

Overall, Figure 6 illustrates that across combinations of both types of covariate data and outcomes, the clustering methods demonstrate remarkable improvement over the merging approaches. Figure 6A shows the results from the molecular profiling data, in which we observe a median improvement of 20-30% of the ensembles over the merged-based learners when predicting the number of mutations. This suggests that the variable selection and model-based clustering strategy in `vscc` is highly effective for construction of ensembles. Figure 6b highlights the impressive improvement of the *Sample Weighted* and *Stack Ridge* over the *Merged* when considering gene expression data and the continuous mutation count outcome, with median percent changes around or above 75%. As described in the original paper,

the clusters previously determined through nonparametric methods were found to align with biologically relevant characteristics [23]. Interestingly, we observe across all scenarios that while the *Stack Ridge* does improve upon the *Sample Weighted*, the difference between the two is fairly marginal. This indicates that there are settings in which the choice of weighting scheme does not drastically influence the prediction ability of the ensemble; however, even in this case, the *Stack Ridge* proves optimal, albeit by a lower margin than in many of our other demonstrative examples. In 6c, we see a greater augmentation of the *Stack Ridge* over the *Sample Weighted*, with decreased but still significant improvements over the *Merged*. Overall, these results display the remarkable robustness of the CCWF method to different types of outcomes, covariate data, and cluster construction, all in a real-life biological setting.

Finally, we explore the influence of variable importance measures on ensemble performance. We posit that the CCWF approach is most effective when the variables that determine the clusters are not also the variables most associated with the outcome. Otherwise, the true outcome model is obscured within each cluster and the cluster-specific learners are unlikely to generalize. For the molecular profiling data, we experimented with only training ensembles on the variables chosen to cluster over, and obtained significantly worse overall performance. To check whether our proposed condition holds within this dataset, we examined the variable importance rankings from the *Subset Merged* of the variables chosen to cluster over. The Random Forest algorithm computes the importance of each variable in accurately predicting the outcome in training; these metrics can be used to categorize which variables are most essential and those that have less influence [26]. We found that there was a median of 20% overlap between the two sets (IQR 0 - 33%); this fairly low level supports our hypothesis that the variables determining the clustering are not important in predicting the outcome within this particular dataset. We additionally experimented with only training ensembles on the variables chosen to cluster over within the molecular profiling data, and obtained significantly worse overall performance. These findings provide insight into when ensembling over clusters is a better strategy over merging.

5 Conclusions

Overall, we have demonstrated that building ensembles of forests trained on clusters determined by k-means or other model-based clustering algorithms results in generalizable and accurate predictors. Such ensembles are robust to changes in the covariate-outcome model as well as the structure and distribution of the covariate data. We improve upon the traditional Random Forest algorithm with equivalent total numbers of trees when the data is naturally divisible by clustering methods. Biological datasets (among several other applications) often contain heterogeneity in the distribution of the features, so we posit that the CCWF method may be potentially useful across a variety of settings. We additionally illustrate the utility of this general framework when dealing with multiple datasets, in which we improve upon the traditional multi-study paradigm.

The results of our simulations and data application provide insight into the importance of data partitioning for the Random Forest algorithm. We have demonstrated that it is not necessarily optimal to divide the data based on true clusters (or studies, for the multi-study setting). Interestingly, reducing within-cluster covariate heterogeneity allows the forests learn

the covariate-outcome rule more efficiently even though the sample size may be greatly reduced from that of the total dataset. Finally, we have established that a CCWF has the ability to equal the performance of a complex Neural Net even when the traditional Random Forest algorithm does not.

6 Future Directions

There are several possible directions to extend this research in the future. We have initiated theoretical analysis of ensembling over clusters for both Random Forest and linear regression, and preliminary results support the conclusions of this paper.

However, we have not yet incorporated the k-means clustering step into the analysis, as such addition greatly increases the complexity of mathematically representing the predictions of the ensemble. Furthermore, the utility of weighting each individual tree in the ensembles through stacking weights instead of simply weighting the forests should be explored further. Ramchandran et. al. (2020) have previously demonstrated this paradigm to be highly effective within the multi-study setting, and Supplementary Figure S2 shows that similarly, weighting trees within ensembles built on clusters produces improvements compared to the strategy of weighting forests [7]. We chose to limit the scope of the analysis within this paper to weighting forests in order to characterize the general strategy more clearly and present an analytically tractable and generalizable framework. However, future work could more closely examine the effect of individually weighting the trees within ensembles and determine whether the conclusions of this paper still hold in that case.

7 Acknowledgments

We thank Prasad Patil for his valuable insights and suggestions. Maya Ramchandran was supported by NIH-NCI Training Grant T32CA009337. Rajarshi Mukherjee was partially supported by NSF Grant EAGER-1941419. Giovanni Parmigiani was supported by grants NSF-DMS 1810829 and NIH-NCI 4P30CA006516-51.

References

[1] Geert Verbeke and Emmanuel Lesaffre. A linear mixed-effects model with heterogeneity in the random-effects population. *Journal of the American Statistical Association*, 91(433):217–221, 1996.

[2] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[3] Ran Avnimelech and Nathan Intrator. Boosted mixture of experts: An ensemble learning scheme. *Neural Comput.*, 11(2):483–497, February 1999.

[4] Prasad Patil and Giovanni Parmigiani. Training replicable predictors in multiple studies. *Proceedings of the National Academy of Sciences*, 115(11):2578–2583, 2018.

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

[6] Gerard Biau and Erwan Scornet. A random forest guided tour. *TEST*, 25(2):197–227, October 2016.

[7] Maya Ramchandran, Prasad Patil, and Giovanni Parmigiani. Tree-weighting for multi-study ensemble learners. *Pacific Symposium on Biocomputing*, 25:451–462, 2020.

[8] Meghana Deodhar and Joydeep Ghosh. A framework for simultaneous co-clustering and learning from complex data. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, page 250–259, New York, NY, USA, 2007. Association for Computing Machinery.

[9] Robert Schapire. The boosting approach to machine learning: An overview. *Nonlinear Estimation and Classification. Lecture Notes in Statistics*, 171:5–32, 2003.

[10] Thomas G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, page 1–15, Berlin, Heidelberg, 2000. Springer-Verlag.

[11] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996.

[12] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[13] Donghui Yan, Aiyou Chen, and Michael I. Jordan. Cluster forests. *Computational Statistics and Data Analysis*, 66:178–192, 2013.

[14] Tao Shi and Steve Horvath. Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics*, 15(1):118–138, 2006.

[15] M. Bicego. K-random forests: a k-means style algorithm for random forest clustering. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[16] Shubhendu Trivedi, Zachary A. Pardos, and Neil T. Heffernan. The utility of clustering in prediction tasks. *CoRR*, abs/1509.06163, 2015.

[17] Leo Breiman. Stacked regressions. *Machine Learning*, 24(1):49–64, July 1996.

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[19] Weiliang Qiu and Harry Joe. clustergeneration: Random cluster generation (with specified degree of separation). *R*, 2020.

[20] Niels G. Waller. *fungible: Psychometric Functions from the Waller Lab.*, 2020. version 1.95.4.8.

[21] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38(4):367–378, 2002. Nonlinear Methods and Data Mining.

[22] Benjamin Frederick Ganzfried, Markus Riester, Benjamin Haibe-Kains, Thomas Risch, Svitlana Tyekucheva, Ina Jazic, Xin Victoria Wang, Mahnaz Ahmadifar, Michael J Birrer, Giovanni Parmigiani, Curtis Huttenhower, and Levi Waldron. curatedOvarianData: clinically annotated data for the ovarian cancer transcriptome. *Database (Oxford)*, 2013:bat013, 2013. PMCID: PMC3625954.

[23] Cancer Genome Atlas Research Network. Comprehensive, integrative genomic analysis of diffuse lower-grade gliomas. *The New England journal of medicine*, 372(26):2481–2498, 2015.

[24] Jeffrey L. Andrews and Paul D. McNicholas. Variable selection for clustering and classification. *arXiv preprint*, 2013. arXiv:1303.5294.

[25] Luca Scrucca, Michael Fop, T. Brendan Murphy, and Adrian E. Raftery. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, 8(1):289–317, 2016.

[26] Kellie J. Archer and Ryan V. Kimes. Empirical characterization of random forest variable importance measures. *Computational Statistics and Data Analysis*, 52(4):2249–2260, 2008.