Conditional Super Learner

Preprint · December 2019

CITATIONS
0

READS
150

3 authors, including:

Gilmer Valdes
University of Pennsylvania
76 PUBLICATIONS **854** CITATIONS
SEE PROFILE

Yannet Interian
University of San Francisco
35 PUBLICATIONS **323** CITATIONS
SEE PROFILE

Conditionally Interpretable Super Learner

Gilmer Valdes*, Yannet Interian*, Efstathios Gennatas and Mark Van der Laan

Abstract—In this article we consider the Conditional Super Learner (CSL) algorithm that conditional on the covariates, selects the best model candidate from a library of models . The CSL expands the idea of using cross validation to select the best model and merges it with meta learning. Here we propose a specific algorithm that finds a local minimum to the problem posed, proof that it converges at a rate faster than $O_p(n^{-1/4})$ and offer extensive empirical evidence that it is an excellent candidate to substitute stacking or for the analysis of Hierarchical problems. Additionally, implication for global interpretability and model distillation are also emphasized.

Index Terms—Cross-validation; Meta Learning; Super learner; Interpretability; Non Parametric Hierarchical models.

1 Introduction and Related Work

The idea of combining different models to obtain one that is better than any of its constituents (meta learning) has been explored extensively and it is currently used in many applications [1]–[3]. Meta learning today, however, mainly consists of creating linear combinations of models (i.e stacking). Its purpose is to improve the accuracy of the individual models, albeit at the expense of interpretability. Related ideas are also explored for ensemble methods which create linear combination of simpler models. Two main ensemble methods can be highlighted: bagging and boosting [4], [5]. In bagging, models are averaged to reduce the variance of individual models and improve accuracy. In boosting, simple models are sequentially learned reducing the bias of the estimator at each step [6].

Usually thought independently from meta learning, the use of cross validation to select the best algorithm from a library (either different models or different hyperparameters) is widely popular [7]. Establishing the theoretical basis for designing an oracle algorithm that will select the best from a library of models (using cross validation), Van der Laan et al demonstrated that cross validation can be used more aggressively than previously thought, terming the cross validation selector "super learner" [8]. Specifically, it was shown that if the number of candidate estimators, $K(n)$, is polynomial in sample size, then the cross validation selector is asymptotically equivalent to the oracle selector –one that knows the best algorithm [8]. Similarly to the empirical use of cross validation, the super learner proposes to select one model from the library for all the observations. Everything else being equal, we would prefer if a simple model is selected to be able to afford interpretability and easier deployment. However, if the data generating process is complex and one uses simple models in the library, it is possible that the model selected to be the best in one region of the covariates might not be the best in another. Therefore, using simple models in the library will introduce biases and decrease accuracy.

In the present article, we develop an algorithm that selects the best model from a library conditional on the covariates, called here the Conditional Super Learner (CSL). This meta algorithm can be thought as learning in the cross validation space. With the CSL, therefore, we investigate a meta learning strategy that reduces the bias of the models in the library by selecting them conditional on the covariates. We show how the CSL has implications for both the accuracy of models and their interpretability. Additionally, we also provide extensive empirical results on how the CSL can be seen as an alternative to stacking and due to its hierarchical nature, an excellent candidate for the analysis of hierarchical data. Specifically, in this article we:

1) Develop the theoretical foundations for the *Conditional Super Learner*: An algorithm that selects the best model from a library conditional on the covariates.
2) Provide convergence rate theorems and oracle inequalities for the CSL.
3) Illustrate how the CSL is a generalized partitioning algorithm that finds different boundary functions (not just vertical cuts as CART does) with $\mathcal{M}$-estimators algorithms at the nodes.
4) Establish the connection between CSL and interpretability.
5) Show empirically how CSL improves over the regular strategy of using cross validation to select the best model for all observations.
6) Show empirically how CSL can give better R^2 than stacking in a set of regression problems.
7) Show empirically how CSL performs in the analysis of Hierarchical Data.

2 Conditional Super Learner

The algorithms that we discuss in this paper are supervised learning algorithms. The data are a finite set of paired observations $\mathcal{X} = \{(x_i, y_i)\}_1^N$. The input vectors x, whose

- *G. Valdes is with the Department of Radiation Oncology, University of California San Francisco, San Francisco and with the Department of Epidemiology and Biostatistics, University of California San Francisco, San Francisco, CA. E-mail: Gilmer.Valdes@ucsf.edu*
- *Y.Interian is with the M.S. in Data Science at University of San Francisco, San Francisco, CA*
- *E. Gennatas is with Department of Radiation Oncology, Stanford University, Palo Alto, CA*
- *M. Van der Laan is with the Division of Biostatistics, University of California, Berkeley, CA*
- **First authors*

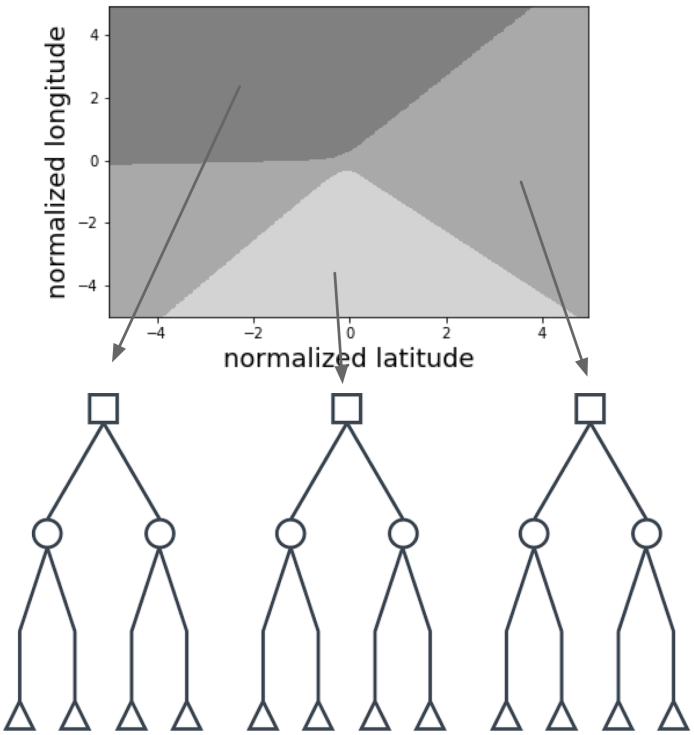

Fig. 1: This diagram shows an application of the CLS model. In this dataset we have 4 variables: number of bedrooms, bathrooms, latitude and longitude to predict house prices. The rectangular region shows how the oracle divides the latitude and longitude (normalized) in 3 regions. Each region has its own expert (using number of bedrooms and bathrooms), represented here by a diagram of a tree, to makes predictions.

components are called here covariates, are assumed to be on $\mathbb{R}^p$, while y can be either a regression or classification label.

We propose to solve supervised learning problems by 1) dividing the input space into a set of regions that are learned by an iterative algorithm and 2) fit simple interpretable models that we call “experts" to the data that fall in these regions. The regions are learned by fitting a multi-class classification model that we call the “oracle" which learns which expert should be used on each region. Given an oracle $o(x)$, region k is defined as $\{o(x) = k\}$, that is, the set of points for with the oracle predict to use the function $f_k(x)$.

An example of an application of the CSL is shown in Figure 1. In here we have 4 variables to predict houses prices: bedrooms, bathrooms, latitude, longitude. The rectangular region shows how the oracle divides the latitude and longitude (normalized) in 3 regions. Each region has its own expert, represented here by a diagram of a tree, to make predictions. Each of the experts has as input the 4 variables.

As with any meta-learning algorithm, the Conditional Super Learner algorithm for learning the oracle $o(x)$ (given the fits of the K experts) will be applied to a cross-validated data set, using V-fold cross validation. That is, for each Y_i falling in one of the V validation samples, we have a corresponding training sample. We couple each observation Y_i with K expert algorithms trained on subsets (from current best estimate of oracle) of its corresponding training data set, thereby creating a cross-validated data set of N observations. In this section, for the sake of explaining the conditional super-learner algorithm, this formality of cross validation will be suppressed, but in our theoretical section (see supplementary material) we make the full conditional super-learning algorithm formal.

2.1 Definition of CSL

Given an oracle $o(x)$ and K experts models $\{F_k(x)\}_{k=1}^K$ fitted on each of the corresponding regions $\{o(x) = k\}$, the CSL can be defined as:

$$CSL(x) = \sum_{k=1}^{K} \mathbb{1}\{o(x) = k\} F_k(x) \tag{1}$$

where $o(x) \in \{1, 2, ..K\}$. $CSL(x)$ is the Conditional Super Learner that outputs the prediction from the best model $F_k(x)$ selected from a library of K models conditional on the covariate x. The idea is to find the $o(x)$ and corresponding fits $\{F_k(x)\}_1^K$ that minimize a given loss function over the training data:

$$\underset{o,\{F_k\}_1^K}{\operatorname{argmin}} \sum_{i=1}^{N} L\big(y_i, \sum_{k=1}^{K} \mathbb{1}\{o(x_i) = k\} F_k(x_i)\big) \tag{2}$$

2.2 Fitting the oracle

To find the solution to equation 2 we will employ a trick often used in machine learning. We will iterate between solving $o(x)$ and solving for $\{F_k(x)\}_1^K$. To solve for $o(x)$ we will assume that all $\{F_k(x)\}_1^K$ are known, the library, and that we also have unbiased estimations (i.e., cross-validated) of the loss at each training point $L(y_i, F_k(x_i))$. In this case, $CSL(x)$ will aim to find the best $o(x)$ that minimizes the loss function over the training data

$$\underset{o(x)}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i, \sum_{k=1}^{K} \mathbb{1}\{o(x_i) = k\} F_k\left(x_i\right)\right) \tag{3}$$

and using the definition of the indicator function, we can take the sum outside of the loss function and get Equation 4:

$$\underset{o(x)}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}\{o(x_i) = k\} L(y_i, F_k(x_i)) \tag{4}$$

To introduce how we fit the oracle, we define a new dataset that we called “extended" dataset. This is a dataset for a multi-class classification problem with K classes, each class corresponding to one of the expert models. This dataset has $K \cdot N$ observations – each x_i appears K times with corresponding labels $\{1, \dots K\}$ and a specific weight.

Definition 1. *Given dataset $\mathcal{X} = \{(x_i, y_i)\}_1^N$, expert functions $\mathcal{F} = \{F_k\}_1^K$ we define the* **extended dataset** *$\mathcal{X}_\mathcal{F} = \{(\tilde{x}_i, z_i, w_i)\}_1^{K \cdot N}$ with $K \cdot N$ observations where:*

- $\tilde{x}_i = x_{\lfloor i/K \rfloor + 1}$
- $z_i = i \mod K + 1$
- *w_i is a weight on observation $(\tilde{x}_i, z_i)$ and is defined in the following way:*

Let l_i be K dimensional vector $l_i = (L(y_i, F_1(x_i)), \ldots, L(y_i, F_K(x_i))$ where element k is the loss of expert k at point (x_i, y_i). Let ONE_K a $K \times K$ matrix of all ones and $DIAG_K$ a $K \times K$ matrix with ones in the diagonal and zeros everywhere else.

$$(w_{iK+1}, \ldots w_{iK+K})^T = [ONE_K - DIAG_K]^{-1} l_i^T \quad (5)$$

Lemma 2.1. *Solving problem 4 is equivalent to finding the oracle $o(x)$ that minimizes the weighted miss classification error of a extended dataset $\mathcal{X}_\mathcal{F}$:*

$$\underset{o(x)}{\operatorname{argmin}} \sum_{i=1}^{N \cdot K} w_i \mathbb{1}\{o(\tilde{x_i}) \neq z_i\} \quad (6)$$

Proof. First note that a missclassification loss for a multi-class classification problem can be written as $L(y, f(x)) = \mathbb{1}\{f(x) \neq y\}$. We want to write Equation 4 as a missclassification loss of a classification problem. For each observation (x_i, y_i) consider the weighted dataset $\{(x_i, 1, w_1), (x_i, 2, w_2), \ldots (x_i, K, w_K)\}$ with missclassification loss $\sum_{k=1}^{K} w_k \mathbb{1}\{o(x_i) \neq k\}$. That is, we want to find for each observation (x_i, y_i) weights $(w_1, \ldots, w_K)$ such that:

$$\sum_{k=1}^{K} \mathbb{1}\{o(x_i) = k\} L(y_i, F_k(x_i)) = \sum_{k=1}^{K} w_k \mathbb{1}\{o(x_i) \neq k\} \quad (7)$$

Since $o(x_i)$ can just have values in $\{1, \ldots, K\}$ we can consider all the options. For example, if $o(x_i) = k$, the equality in Equation 7 becomes $L(y_i, F_k(x_i)) = \sum_{j=1}^{K} w_j - w_k$. If we consider all possible values for $o(x_i)$ we get the following set of equations:

$$\begin{aligned} L(y_i, F_1(x_i)) &= \sum_{j=1}^{K} w_j - w_1 \\ &\vdots \\ L(y_i, F_K(x_i)) &= \sum_{j=1}^{K} w_j - w_K \end{aligned}$$

The previous equation can be written in matrix form $l_i^T = [ONE_K - DIAG_K](w_1, \ldots w_K)^T$. Which gives us:

$$(w_1, \ldots, w_K)^T = [ONE_K - DIAG_K]^{-1} l_i^T$$

□

As a result of Lemma 2.1, $o(x)$ is the solution of a multi-class classification problem on the extended dataset $\mathcal{X}_\mathcal{F}$. We approximate o(x) by fitting any standard classification algorithm on $\mathcal{X}_\mathcal{F}$.

2.3 Fitting the experts

Similarly to the previous section, in order to fit the experts we assume that $o(x)$ is known. Then Equation 2 becomes K independent classification/ regression problems that minimize the empirical loss over observations $\{i : o(i) = k\}$, for each $k = 1, \ldots, K$, which is generally already solved by standard machine learning algorithms.

$$\underset{F_k}{\operatorname{argmin}} \sum_{x_i : o(x_i) = k} L(y_i, F_k(x_i)) \quad (8)$$

Algorithm 1: Conditional Super Learner **(CSL)**

Input: $\mathcal{X} = \{(x_i, y_i)\}_1^N$; $\mathcal{F} = (F_1, F_2, \ldots, F_K)$
Initialize: for each sample split $v = 1, \ldots, V$, fit the experts $\mathcal{F} = (F_1, F_2, \ldots, F_K)$ on initial subsets of the v-th training data set. For each i, let $F_{k,-i}$ be the k-th expert trained on training sample that excludes Y_i. Construct the corresponding cross-validated data set $(Y_i, F_{1,-i}(x_i), \ldots, F_{K,-i}(x_i))$, $i = 1, \ldots, N$.
for $t = 1$ *: iterations* **do**
  For each point and each expert compute: $L(y_i, F_{k,-i}(x_i))$
  Create extended dataset $\mathcal{X}_\mathcal{F}$
  Fit $o(x)$ on $\mathcal{X}_\mathcal{F}$
  Re-fit each expert F_k on $\{o(x) = k\}$ for the V-training samples.
end
Based on final $o(x)$, refit each expert F_k on $\{o(x) = k\}$ for total sample.
Result: $\sum_{k=1}^{K} \mathbb{1}\{o(x) = k\} F_k(x)$

2.4 A two step algorithm

Finding $\{F_k(x)\}_1^K$ indicates that equation 2 can be minimized iteratively. Following the K-mean algorithm's philosophy, let us propose the minimization of equation 2 in two steps: one to fit the oracle and the other to fit the experts. Please note that if we take into consideration that at every time that each step is applied the Loss function decreases, the convergence to a local minimum is guaranteed. Of course, this is only true if we use, for each observation, the estimation of $(y_i, F_k(\mathbf{x_i}))$ on the training data. This, however, will most likely result in overfitting. After this discussion we are ready to write Conditional Super Learner pseudo code (see above).

3 Simulations

In this section we describe our experiments that serve to benchmark the CSL and illustrate where it is most useful. In our empirical analysis, we considered all regression problems from the Penn Machine Learning Benchmarks [9]. In the first set of experiments, datasets where the number of observations was between 200 and 500000 ($N = 84$) were considered. In the second set of experiments, we selected a subset of those with at least 2000 points ($N = 19$). We report R^2 as the performance metric. In the first experiments, datasets were split in 80% training and 20% testing sets. For the second experiment, we need a validation set, therefore data was split in $70\%/15\%/15\%$ for train/validation/test.

In all our experiments we use the following set of base algorithms or experts:

1) Ridge: alphas = [1e-4, 1e-3, 1e-2, 1e-1, 1, 2, 4, 8, 16, 32, 64, 132]
2) ElasticNet: l1 ratio = 0 (Lasso)

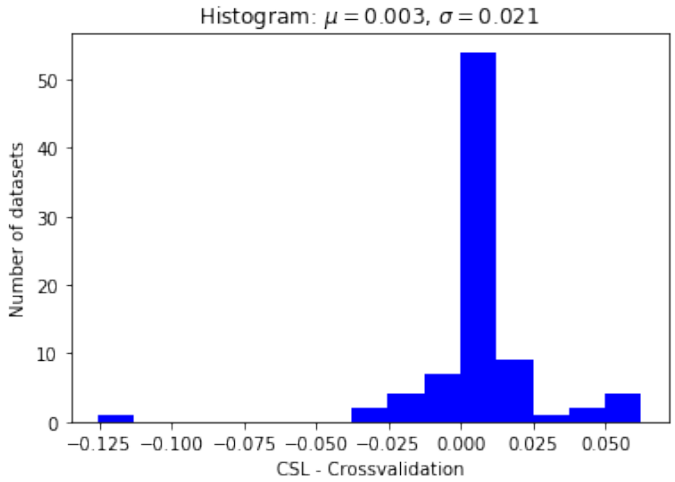

Fig. 2: One shot CSL vs cross validation. Partitioning the space in two regions with a depth = 1 decision tree and choosing a model for each results in an improvement, on average, over cross-validaiton. Overfitting, however, is still possible.

3) ElasticNet: l1 ratio = 0.5
4) Decision Tree: max depth = 4
5) Decision Tree: max depth = 5
6) Decision Tree: max depth = 6

Although the CSL can be used with more complex models as experts, using simple linear or tree models have two appealing: protects the algorithm for overfitting and has implications for its interpretability as discussed below.

3.1 Single step CSL and its comparisson to cross validation

First, we wanted to evaluate empirically how CSL ($\mathcal{F}_3$ class of meta learners in the theoretical section presented in the supplemental material) performs compared to the naive strategy of using the expert model selected through cross validation for all the points ($\mathcal{F}_1$ class in the theoretical section). For this we used the library of experts described above and a decision tree algorithm with $max\_depth = 1$ as our oracle. No partition was allowed if the terminal node did not have more than 2% of observations belonging to it. This decision tree algorithm was selected as the oracle because if not partition is performed, then the minimization of equation 1 results in selecting the model that minimizes the cross validation error (equal to $\mathcal{F}_1$ class in the supplemental material). As such, our CSL in this case includes the possibility of just using cross validation and it should perform, on average, better than using cross validation to select one model. Please note that in this section the experts are obtained on all the training data and only one iteration is allowed for the meta. The empirical evaluation of this CSL was compared to the performance obtained if we use cross validation to select the best model from one of the sixth algorithms mentioned above.

Figure 2 shows the results obtained when we compared R^2 for both CSL and cross validation. In 77.5% of the datasets CSL had at least the same performance as cross validation and in 20% its R^2 was bigger by at least 1%. Although these results show that CSL improves over naive cross validation, overfitting can happen and extra attention needs to be paid. In fact, we obtained an outlier where cross validation outperformed the CSL by 0.125. This problem is the synthetic dataset $658 - fri - c3 - 250 - 25$ from the Friedman's regression datasets [9], [10]. The same is a hard problem where algorithms are prompt to overfit since training only contains 200 points with 25 explanatory variables, several of them correlated to each other. Additionally, for those datasets where CSL was better, on average they had 3805 data points compared to 2368 in those datasets where cross validation did better. Therefore, as a general rule, CSL needs more than 2500 data points to perform better than cross validation. These empirical results corroborate our suggestions on the theoretical session and the need for a two tier Super Learner algorithm where the type of meta strategy, or class of functions as defined here, is also selected.

In the next section, we will compare a full CSL to stacking. $\mathcal{F}_2$ vs $\mathcal{F}_3$ meta learning strategies in our theoretical discussion (supplemental material).

3.2 CSL versus Stacking

Stacking [1], [3], [11] is a general procedure where a learner is trained to combine the individual learners. The base level models are trained on the original training set, then a meta-model is trained on the outputs of the base level model as features. The base level often consists of different learning algorithms. In our experiments, we use the same 6 base models defined in previous section. As meta-model in stacking we use a linear regression model.

CSL. In this experiment, we use the same set of experts as in previous section and we use a two layer feed-forward neural network as the oracle. The oracle was written in PyTorch and fitted with Adam optimizer with learning rates of 0.15. The number of epochs at each iteration was a function of the sample size ($\frac{3000}{\log(N)^2}$). The hidden layer was also set as a function of the sample size ($\min(2\log(N), 150)$). After the first linear layer and before the Relu activation function, batch normalization was used. A dropout layer with $p = 0.2$ is used before the second linear layer.

In Table 1 we show results from comparing CSL and stacking on regression datasets. For each dataset we run each algorithm 10 times by spliting training, validation and testing sets using different seed. *CSL_mean* shows the mean R^2 over all experiments. Similarly, *Stack_mean* shows the mean R^2 of the stacking experiments. *Diff* show the difference between *CSL_mean* and *Stack_mean*. Column *Test results* shows whether a t-test found the difference in mean to be significant. There were 19 datasets in this experiment. For one dataset stacking is significantly better than CSL. In 8 problems both algorithms are statistically the same. In 11 problem CSL is better. In the experiment for dataset *574_house_16H*, one of the runs produces an outlier which is responsible for the mean difference.

3.3 CSL on hierarchical synthetic data

Hierarchical models are extremely important in fields like Medicine [12]. CSL, due to its architecture, seems specially suited to analyze hierachical data. In this section, we evaluate

Dataset	CSL_mean	Stack_mean	Diff	Test result
1193_BNG_lowbwt	0.623	0.594	0.028	non-significant
1199_BNG_echoMonths	0.532	0.452	**0.080**	significant
1201_BNG_breastTumor	0.145	0.109	**0.036**	significant
1203_BNG_pwLinear	0.619	0.606	**0.013**	significant
197_cpu_act	0.977	0.959	**0.018**	significant
201_pol	0.964	0.908	**0.056**	significant
215_2dplanes	0.943	0.929	**0.014**	significant
218_house_8L	0.541	0.574	-0.033	significant
225_puma8NH	0.631	0.608	**0.023**	significant
227_cpu_small	0.964	0.955	0.009	non-significant
294_satellite_image	0.804	0.809	-0.005	non-significant
344_mv	0.992	0.979	**0.012**	significant
503_wind	0.772	0.756	**0.016**	significant
529_pollen	0.789	0.790	-0.000	non-significant
537_houses	0.652	0.661	-0.009	non-significant
562_cpu_small	0.965	0.955	0.011	non-significant
564_fried	0.900	0.775	**0.126**	significant
573_cpu_act	0.973	0.961	**0.012**	significant
574_house_16H	0.167	0.443	-0.276	non-significant

TABLE 1: Results comparing CSL and stacking on 19 regression datasets. In one dataset stacking is significantly better than CSL. In 8 problems both algorithms are statistically the same. In 11 problem CSL is better.

Dataset	CSL	Base	RF	GBM
564_fried	**0.97** (0.02)	0.45 (0.17)	0.82 (0.04)	0.82 (0.06)
574_house_16H	**0.98** (0.01)	0.84 (0.04)	0.87 (0.01)	0.93 (0.01)
294_satellite	**0.99** (0.01)	0.88 (0.01)	0.98 (0.01)	0.98 (0.01)
218_house_8L	**0.97** (0.02)	0.75 (0.03)	0.94 (0.01)	0.93 (0.01)

TABLE 2: Results on running CSL on synthetic data. Comparison are made with random forest, gradient boosting and the best preforming base expert. For each dataset the we generate 10 synthetic problems. For each method, the mean test R^2 and standard deviation is shown.

the performance of CSL on hierarchical synthetic problems. Specifically, we wanted to investigate if:

1) CSL can discover hierarchical structures.
2) Compare its performance in these type of problems to gradient boosting and random forest.

Given a real dataset $\{(x_i, y_i)\}_1^N$ from the Penn Machine Learning Benchmarks, we generate syntetic data by using the observations from the covariates but generating new labels. Here is how we generated the new ($\tilde{y}$):

1) Sample 70% of the data.
2) Use the covariates to find $K = 3$ clusters using K-means algorithm.
3) Fit a 2-layer neural network using the cluster id as a label.
4) Using the neural network, predict a cluster id (l_i) on each of the original observations, creating the dataset $\{(x_i, y_i, l_i)\}_1^N$.
5) Fit each subset $\{(x_i, y_i)$ such that $l_i = k\}$ for $k \in \{1, 2, 3\}$ to a regression model (Ridge). Use the prediction from the regression model as the new synthetic label ($\tilde{y}$).

For the experiments in Table 2, gradient boosting was ran with the following parameters: min_child_weight=50, learning rate = 0.1, colsample_bytree= 0.3, max_depth= 15, subsample=0.8, and with 500 trees. For random forest, we used 1000 trees, max_features='sqrt' and we found max_depth with cross validation for each problem. The problems are a subset of the problems in Table 1, where the R^2 of the base expert is lower than 0.9. The CLS algorithm was initialized using 11 linear models as based experts. It turns out that by using trees together with linear models on these problems the algorithm sometimes would get stuck in suboptimal local minima so run the CSL multiple times.

Table 2 shows the results (R^2) of CSL on synthetic data compared to random forest, gradient boosting or the best base expert (selected using cross validation). As it can be seen, CSL significantly outperforms all other algorithms as expected.

3.4 Implications on interpretability

In many high-stake domains like medicine and the law, interpretability of machine learning algorithms is highly desirable since errors can have dire consequences [13]–[15]. This fact has led to a renew interest for the development of interpretable models. Interpretability, however, can only be judge relative to the field of application as it is in the eyes of the beholders. In fact, there is a considerable disagreement on what the concept means and how to measure it [16]–[18]. Different algorithms afford different degrees of interpretability and even black boxes can be investigated to gain some intuition on how predictions are being made. For instance, variable importance or distillation can be used to interpret neural networks [19]. This level of interpretability might be enough for applications that do not impose high risk. In other applications (e.g medicine), the need to understand the models globally rises [13]. Without attempting to formally quantify and define interpretability here, we will illustrate below how the CSL results in models that are highly transparent.

Predicting house prices. To illustrate how CSL can be use as an interpretable algorithm we use a dataset of house rental prices in New York City. We have 4 input variables: latitude, longitude, number of bedrooms and number of bathrooms. Two make it really simple to visualize and interpet, the oracle was given two of the variables: latitude and longitude. The CSL model found a solution in which the oracle parition the space of latitude and longitude in 3 regions (see top of

Figure 1) and for each region a tree of depth 5 predicts the house prices. This simple solution get an R^2 of 0.68. As a comparison, the best single model of a tree of depth 5 has an R^2 of 0.62 and a random forest with 500 trees (of depth 9) has an R^2 of 0.72. To find the best random forest we did grid search on the number of variables and the depth of the trees.

The simple solution of a tree of depth 5 is interpretable since a tree of depth 5 can be easily examined. Also, 3 trees of depth 5 can be easily examined as well as the 2 dimensional space where the oracle split the restricted input regions. On the other hand, the random forest with 500 trees and unrestricted depth would be harder to interpret.

4 Theoretical Results

In this section, due to space constrains, we will summarize the main theoretical results obtained and presented in the supplemental materials. The main result, proven on Theorem S1, establishes that a Super Learner given by the function f_n will converge to the oracle f_{0n}, the true underlying function generator, at a rate at least faster than $O_p(n^{-1/4})$ being n the number of observations. Additionally, we provided a theoretical analysis of the variance-bias tradeoff incur for different choices of the meta learning algorithm and the base learners. Specifically, if one considers relatively large meta-learning models, as is easily the case for our conditional super-learner, then there is a risk that one worsens the performance relative to simpler meta-learning models. Therefore, the most sensible strategy is to define a sequence of Super Learner models where one goes from the simplest (cross-validation) to more complex (trees, neural networks, etc). This process is referred as double super-learning and a detailed discussion is presented in the supplemental materials. We then finalized by presenting an oracle inequality for the double super-learner showing that it is asymptotically equivalent to a super-learner that knows the true oracle choice of meta-learning. Additionally, we also presented a finite sample oracle inequality for an ϵ-net double super-learner. These results establish that the double super-learner also approximates the oracle choice in its meta-learning model at a rate at least as fast as $n^{-1/4}$.We refer those readers interested in diving deeper into our theoretical analysis to the supplemental materials.

5 Discussion

In the present article we introduced the CSL, provided extensive empirical simulations and baseline results in a wide variety of problems and mathematical proofs about its convergence rates (see supplemental material). Additionally, to further the understanding of the CSL, let us highlight different relationships and connections that it has with different algorithms. First, please note that $o(x)$ partitions the space in K regions or subsets $\{\mathcal{R}\}_1^K$ where the models $\{F_k(x)\}_1^K$ are used for prediction; ergo establishing the connection between meta learning and generalized partitioning machines through the CSL. Different from recursive algorithms like CART, MediBoost or the Additive Tree [13], [20], [21], CSL partitions defined by the oracle can have complex forms and are not forced to be perpendicular to the covariates. Additionally, $CSL(x)$ also generalizes the strategy of using cross validation to select the best model. Please note that if in equation 1 we force $o(\mathbf{x}) = \mathbf{c}$ where c is a constant $\in \{1..K\}$ then the solution to 4, $\hat{o}(\mathbf{x})$, just selects the model that minimizes the cross validation error. As such, using cross validation to select the best model is the simplest case of $CSL(x)$ where the meta learner predicts a constant regardless of the covariates. CSL can also be thought as a generalization of the K-means algorithm. If the expert models are constant, and the oracle has infinity capacity to always be able to assign each observation to the best mean, then CSL becomes the K-mean algorithm. Another algorithm that is closely related to the CSL is the Hierarchical Mixture of Experts(HME) [22]. In both cases, a hierarchical topology is found. In the HME, however, this hierarchy depends on a parametric specification of the probability distribution while the CSL can estimate the hierarchical topology in a non parametric fashion with a rich set of meta algorithms that can include linear models, trees or neural networks. Equally important, the HME predicts with a combination of experts which harms interpretability while the CSL does not. Finally, as shown above, due to its archictecure, CSL is a non parametric hierarchical algorithm and performs quite well for this type of problems. Besides, these theoretical connections with other algorithms, there are some important technical points that can help understand the CSL and point to possible avenues for its improvement. In our experiments, we initialized the CSL by picking the type of experts (e.g trees, linear models, etc) and a random subset of the data to fit each expert to introduce diversity. Please note that as its counterpart (K-means), CSL can get stuck in local minimum. Therefore we ran the algorithm a few times in our experiments and used a validation set (different than the test set) to select the best solution. Researching better initialization methods will likely improve the performance of the algorithm. While running CSL, it is often the case that some of the experts will collapse (e.g model selection). This happens when the range of values predicted by $o(x)$ is less than K or similarly when the size of $\{x : o(x) = k\}$ becomes very small. In these cases we re-adjust the size of K as the algorithm runs. This behaviour is highly desirable because it allows the end user to specify a a large number of bases and the algorithm will provide model selection provided enough regularization is applied. Regarding regularization, there are different ways to proceed with the CSL. One would be to penalize over complex meta learners using the specific hyper-parameters for the chosen model. Additionally, please note that we can use the step when we are fitting the experts to introduce regularization. If the oracle $o(x)$ also estimates probability of an observation belonging to a model (e.g. logistic regression), then we can use $p(0(\mathbf{x}) = \mathbf{k}, \mathbf{x})$ to introduce similarity among the experts when we are fitting them, specially around the boundaries defined by the oracle. Other ways of regularization can also be explored in the future, given that the partition nature of the CSL makes the local models data hungry. In that sense, global regularizations for the local parameters like that explored at the Highly Adaptive Lasso estimators seem to be appealing and will be investigated in the future [23]. Finally, we would like to finish our discussion with some notes on the implications that the CSL can have on interpretability. Post hoc local model explainability is a popular topic today with algorithms like

LIME or SHAP that build models around specific points [24], [25]. Specifically, LIME explains a complex model behaviour for a specific observation by perturbing the covariates around the point, getting the models prediction and then fitting a linear model. Although the specific limitations of this approach to recover the true underlying explanations are beyond the scope of our article, we must say that at its best LIME provides local explanations [25]. For many applications one would like to have global explanation of the models [13], [26]. The CSL is then a hybrid between the idea of using one very simple model for all observations and building a different model for each. By finding a finite number of simple models that will be used for predictions (e.g linear models), it then provides a sense of global understanding of what variables are important and their contribution.

6 Conclusions

In this work we introduced the CSL algorithm. We proved theoretically and empirically how we can extend the idea of meta learning and develop an algorithm that outperforms the naive use of cross validation to select the best model.We proved that the CSL has a rate of convergance faster than $O_p(n^{-1/4})$. More over, we have obtained very interesting and practical results. For instance, CSL outperformed stacking in the datasets analyzed. Additionally, it significantly outperformed Random Forests or Gradient Boosting in the analysis of Hierarchical Data. Finally, its connection to interpretability and other algorithms were highlighted to deepen our understanding of its performance. As such, the CSL is an algorithm suited for the analysis of datasets in high stake domains where hierarchical models, accuracy and intepretability are of paramount importance.

7 Acknowledgments

Research reported in this publication was supported by the National Institute Of Biomedical Imaging And Bioengineering of the National Institutes of Health under Award Number K08EB026500 and by the National Institute of Allergy and Infectious Diseases under Award Number 5R01AI074345-09. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.Finally, we would also like to thank Dr Charles McCullow for initially suggesting to investigate this topic.

References

[1] L. Breiman, "Stacked regressions," *Machine learning*, vol. 24, no. 1, pp. 49–64, 1996.
[2] M. LeBlanc and R. Tibshirani, "Combining estimates in regression and classification," *Journal of the American Statistical Association*, vol. 91, no. 436, pp. 1641–1650, 1996.
[3] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.
[4] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.
[5] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *icml*, vol. 96. Citeseer, 1996, pp. 148–156.
[6] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.
[7] B. Efron, "Estimating the error rate of a prediction rule: improvement on cross-validation," *Journal of the American statistical association*, vol. 78, no. 382, pp. 316–331, 1983.
[8] M. J. Van der Laan, E. C. Polley, and A. E. Hubbard, "Super learner," *Statistical applications in genetics and molecular biology*, vol. 6, no. 1, 2007.
[9] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore, "Pmlb: a large benchmark suite for machine learning evaluation and comparison," *BioData mining*, vol. 10, no. 1, p. 36, 2017.
[10] J. H. Friedman *et al.*, "Multivariate adaptive regression splines," *The annals of statistics*, vol. 19, no. 1, pp. 1–67, 1991.
[11] P. Smyth and D. Wolpert, "Stacked density estimation," in *Proceedings of the 10th International Conference on Neural Information Processing Systems*, ser. NIPS'97. Cambridge, MA, USA: MIT Press, 1997, pp. 668–674. [Online]. Available: http://dl.acm.org/citation.cfm?id=3008904.3008999
[12] S.-L. T. Normand, M. E. Glickman, and C. A. Gatsonis, "Statistical methods for profiling providers of medical care: issues and applications," *Journal of the American Statistical Association*, vol. 92, no. 439, pp. 803–814, 1997.
[13] G. Valdes, J. M. Luna, E. Eaton, C. B. Simone *et al.*, "Mediboost: a patient stratification tool for interpretable decision making in the era of precision medicine," *Scientific Reports*, vol. 6, 2016.
[14] F. Louzada, A. Ara, and G. B. Fernandes, "Classification methods applied to credit scoring: Systematic review and overall comparison," *Surveys in Operations Research and Management Science*, vol. 21, no. 2, pp. 117–134, 2016.
[15] F. Cabitza and G. Banfi, "Machine learning in laboratory medicine: waiting for the flood?" *Clinical Chemistry and Laboratory Medicine*, vol. 56, no. 4, pp. 516–524, 2018.
[16] Z. C. Lipton, "The mythos of model interpretability," *arXiv preprint arXiv:1606.03490*, 2016.
[17] ——, "The doctor just won't accept that!" in *Proc. of Symp. of Interpretable Machine Learning at the Intl. Conf. on Neural Inf. Processing Sys. (NIPS)*, 2017, pp. 1 – 3.
[18] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.
[19] S. Tan, R. Caruana, G. Hooker, and A. Gordo, "Transparent model distillation," *arXiv preprint arXiv:1801.08640*, 2018.
[20] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees. wadsworth & brooks," *Monterey, CA*, 1984.
[21] J. M. Luna, E. D. Gennatas, L. H. Ungar, E. Eaton, E. S. Diffenderfer, S. T. Jensen, C. B. Simone, J. H. Friedman, T. D. Solberg, and G. Valdes, "Building more accurate decision trees with the additive tree," *Proceedings of the National Academy of Sciences*, vol. 116, no. 40, pp. 19 887–19 893, 2019.
[22] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the em algorithm," *Neural computation*, vol. 6, no. 2, pp. 181–214, 1994.
[23] D. Benkeser and M. Van Der Laan, "The highly adaptive lasso estimator," in *2016 IEEE international conference on data science and advanced analytics (DSAA)*. IEEE, 2016, pp. 689–696.
[24] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, 2017, pp. 4765–4774.
[25] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016, pp. 1135–1144.
[26] E. D. Gennatas, J. H. Friedman, L. H. Ungar, R. Pirracchio, E. Eaton, L. G. Reichmann, Y. Interian, J. M. Luna, C. B. Simone, A. Auerbach *et al.*, "Expert-augmented machine learning," *Proceedings of the National Academy of Sciences*, 2020.