Differential Privacy Techniques in Stacked Generalization

Cole Sibbald

August 28, 2020

1 Introduction

Selecting a single best learning algorithm to represent any type of data, continuous or discrete, can prove challenging. The notable case arises when multiple different models offering varying levels of complexity and interpretability are given.

Stacked Generalization offers the ability to develop standardized ways of combining different models together to perform best on the inherited advantages of any single approach. One immediate fear when combining different models is the potential to increase model over-fitting from the learning set.

Differential Privacy is a mathematically rigorous way of providing privacy to individuals within a data set. When viewed from a machine learning perspective, no single "person" (data point) should influence the model, regardless if they are included or not. If over-fitting can be found, the purpose of this research is to attempt to reduce these effects by blending Stacked Generalization and Differential Privacy together.

2 Stacked Generalization

Stacked Generalization was developed as a formal approach to conduct model aggregation of many models for optimal prediction accuracy. (Wolpert, 1992) Whether one is to predict a continuous regressor or a classification the techniques of stacking remain the same. The initial learning set is represented as Θ. $|\Theta| = n$, where each $\theta_i \in \Theta$ is of dimension $\mathbb{R}^{d+p}$, where $d, p \geq 1$ and $i = 1, 2, ..., n$. d represents the dimension of the input space, while p defines the dimension of the output space.

A generalizer $\mathcal{M}_j \in \{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_k\}$ guesses the output given the specified input such that $\mathcal{M}_j : \mathbb{R}^d \rightarrow \mathbb{R}^p$, $j = 1, 2, ..., k$. Once the training of all k separate $\mathcal{M}$ generalizers has been conducted (called Level 0) with some type of Cross Validation (CV), the results will then be fed into the second layer of the system. It is in the level 1 step that the model aggregates over the different level 0 generalizers. A variety of techniques may be used in this situation (surface fitting, averaging, majority vote, etc.). The basic pseudo-code is as follows:

Algorithm 1: Stacked Generalization

Level 0
Train Level 0 Generalizers with Level 0 Input
Level 1 Input = Predict Level 0 Generalizers

Level 1
Train Level 1 Generalizer(s) with Level 1 Input
Level 2 Input = Predict Level 1 Generalizer(s)
...
Repeat process with any number of levels until a final prediction is produced from a single generalizer.

2.1 Level 0 Generalizers

The level 0 generalizers are a set of k different models $\mathcal{M}_j$, $j = 1, 2, ..., k$ that are all trained off of the same level 0 learning set. This training process is generally completed with some type of CV applied, though optimal results are achieved on an ad hoc basis. Once each of the $\mathcal{M}_j$ models has been trained, they will then be used to predict some output. This new output data will be the level 1 learning set used in the second stage of stacking.

2.2 Level 1 Generalizers

Traditional ensemble learning has been known to use a variety of simple techniques to combine model prediction results. Working for both classification and regression responses, methods such as majority vote and prediction averaging have been used with modest results that help leverage the advantages of a collection of different level 0 generalizers.

The novelty around stacked generalization exists in this aggregation step. Instead of using simpler techniques, the prediction results produced in the training phase of level 0, are used as the inputs in the level 1 generalizer. These points are then applied in some non-linear fashion to train weights for each of the level 0 generalizers. Once these k weights (w) are created, a new point may be passed through this network of levels to produce a single prediction, aggregated efficiently over all k level 0 generalizers.

$$\mathcal{C} = \{\sum_{i=1}^{k} w_i \mathcal{M}_i : \text{Constraints}\} \tag{1}$$

In the above equation, $\mathcal{C}$ represents some surface that is fitted. $\mathcal{C}$ is then used as the level 1 generalizer to aggregate level 0 predictions. The idea is to produce a result based on the minimized weights ($\mathbf{w}$) calculated from the level 1 learning set. A variety of constraints may be placed on $\mathbf{w}$ to achieve a desired and interpretable result. When an objective function is minimized this may be seen as the level 1 generalizer.

In theory, many generalizers may be used (such as different objective functions etc.) in this stage to produce another learning set for the next level (level 2), to then be further aggregated for predictions. In all papers reviewed it appears most applications stop at the first level of stacking.

2.3 Additional Topics

2.3.1 Cross Validation

Cross Validation is a technique that separates the learning set Θ into l partitions, $\Theta_i \in \{\Theta_1, \Theta_2, ..., \Theta_l : \cup_{i=1}^{l} \Theta_i = \Theta, \Theta_i \cap \Theta_j = \emptyset, \forall i, j = 1, 2, ..., l, i \neq j\}$. Following this randomized partitioning, a test set and training set are created such that the training set contains $l - 1$ partitions and the test set contains the remaining partition of the learning set ($\Theta = \Theta_{train} \cup \Theta_{test}$). A model $\mathcal{M}$ is trained on the larger training set and then used to predict the outcomes of the test partition. This process will be conducted until every partition in the learning set as been used as test data.

Algorithm 2: Stacked Generalization with Cross Validation

```
### Level 0 ###
for Θ_i in Θ do
    Θ_(−i) = Θ\Θ_i
    for j in 1:k do
        Train M_j with Θ_(−i)
        Predict M_j(Θ_i)
    end
end

### Level 1 ###
Model Aggregation
```

- Leave-one-out

  The original motivating paper (Wolpert, 1992) describes the advantages of leave-one-out (LOO) CV. This computationally intensive approach trains k new generalizers on a training set which consists of the original data with a single element removed (i.e. there are n partitions of the data). Once the training of the generalizers has been completed, the d-dimensional inputs from the single removed element is fed into the system to produce a prediction result. This process is conducted for every data point in the set. The predicted values will then be compared to the known response in the aggregation step (Level 1).

- K-fold

  This is a slightly adapted approach to LOO CV which is more computationally efficient, only separating the data into k partitions (different k than in Stacked Generalization). Reports of improved results by using this approach over the standard LOO Stacked Generalization have been found. (Breiman, 1996)

Algorithm 3: Stacked Generalization with LOO CV

```
### Level 0 ###
for θ_i in Θ do
    Θ_(−i) = Θ\θ_i
    for j in 1:k do
        Train M_j with Θ_(−i)
        Predict M_j(θ_i)
    end
end

### Level 1 ###
Model Aggregation
```

If no CV is used, the stacking algorithm will appear as follows with level 0 being both trained and predicted on the same full learning set.

Algorithm 4: Stacked Generalization with no CV

```
### Level 0 ###
for j in 1:k do
    Train M_j with Θ
    Predict M_j(Θ)
end

### Level 1 ###
Model Aggregation
```

2.3.2 Scoring / Cost Function

The level 1 aggregation step is developed using some specified cost function to establish an optimal fit of models over a convex hull. This cost function is then minimized to establish optimal weights for model combinations. Different cost functions offer different benefits on an ad-hoc basis. The two most studied methods here include *Quadratic* and *Logarithmic* scoring. More techniques and benefits are mentioned by (Yuling, Y. Vehtari, A. Simpson, D. Gelman, A., 2018, 922).

- Quadratic Scoring

  This represents the traditional method of Ordinary Least Squares regression. This method is used regularly with strong results. In (Brieman, 1996, 52) weights are established by minimizing influence on the variance-covariance matrix, created by the predictions of all k generalizers on a single point.

- Log Scoring

  This scoring method yields positive benefits when compared to the method of a Quadratic Loss function. This score is the only "proper local score assuming regularity conditions" of the 5 proposed criterion. (Yuling, Y. Vehtari, A. Simpson, D. Gelman, A., 2018, 922) Quadratic scoring only offers this benifit in select cases.

  This method of scoring gives rise to a number of downsides that can be defined under the term *hypercompression*. Hypercompression will be explored further in section 3.2, though, in summary, Quadratic scoring is preferred due to these limitations.

2.3.3 Constraints

Breiman's paper tests the necessity of assumptions made by Wolpert, specifically $\sum_{i=1}^{k} w_i = 1$ and $w_i \geq 0$ for the weights of the level 1 generalizer. Breiman finds that though both of these constraints are helpful, the necessity of the sum 1 constraint is not so important that results would be significantly different otherwise.

(Ting and Witten, 1999) tested the necessity of the non-negativity constraints proposed in the original stacked generalization foundations (Wolpert, 1992). They found that when using regression response data, these constraints help to increase the accuracy of predictions. Classification data did not need this constraint, however including it still increases model interpretability making it often more desirable in the aggregation phase.

3 A Bayesian Approach

The initial work on stacked generalization works with point estimates. From a Bayesian perspective, the question naturally arises, "How does one combine models together while preserving the posterior densities produced by a variety of generalizers built on separate priors?" This question was originally answered with the usage of Bayesian Model Averaging (BMA). Further exploration lead the way to newly adapted approaches of BMA to be created addressing various issues inherited. This section aims to bridge the gap associated with Bayesian methodology and stacking of predictive means.

3.1 Bayesian Model Averaging

When the data generating model is in the set of level 0 generalizers, BMA is never worse than stacking. (Clarke, 2003) Outside of this extremely fortunate case, BMA can produce terribly inaccurate results putting too much weight on a single model that is not guaranteed to be closest to the true ground truth function. BMA also does not tend to perform well on heteroskedastic data, another challenge in real-world applications.

Variants of BMA

- Pseudo-BMA

  This algorithm utlizes Akaike weights from AIC to aggregate models together. This method has a natural penalty associated with it which prefers less complex models (a leading cause of over-fitting).(Yuling, Y. Vehtari, A. Simpson, D. Gelman, A., 2018, 920)

- Pseudo-BMA+

  Adopted from Pseudo-BMA this algorithm uses an "expected log predictive density" combined with adopted AIC weights to produce better results. (Yuling, Y. Vehtari, A. Simpson, D. Gelman, A., 2018, 920-921)

- SafeBayes

  On heteroskedastic data, SafeBayes has been found to out-perform a standard Bayes approach. These results are based on experiments from (Grünwald and van Ommen, 2017, 1099) with 'in-liers.'

3.2 Stacking of Predictive Distributions

Few changes occur here different than traditional point estimated stacking. The most significant alteration is that instead of combining weights based on the point estimates produced from the level 0 generalizers, these weights are instead used as posterior predictive quantities that the model is minimizing over.

$$\mathcal{C} = \{\sum_{i=1}^{k} w_i \times p(\cdot|\mathcal{M}_i) : \sum_{i=1}^{k} w_i = 1, w_i \geq 0\} \tag{2}$$

The defined score function used remains an important factor when solving this convex optimization problem. Explored in detail by (Grünwald and Van Ommen, 2017), hypercompression is a significant concern when certain scoring rules are chosen.

Logarithmic scoring in particular is not appropriate. The reason for this is due to model space convexity. In the misspecified setting, a log-score may produce misleading results; a collection of bad models may be combined to get seemingly optimal accuracy. In reality, the best model does not have to be included in this collection of weighted models, making the scoring metric non-optimal in this situation. Quadratic scoring does not face this issue and is therefore considered the best scoring solution for the problem.

4 Differential Privacy in Stacked Generalization

Differential Privacy (DP) is a technique that prioritizes the security of each individuals identity. Any database query where an individual's data is used should not be found to influence the overall databases results; whether the person is present in the system or not. In reference to stacking, every individual may be seen as a data point, and a query may be seen as the usage of this data when developing a trained model for inferencing purposes. The influence that any point has on a model should not cause significant influence on the database, no matter how large the training set. This is approximately the definition of over-fitting.

The concerns in stacked generalization is the slow leakage of personal data as multiple level 0 generalizers query the same point to develop their own minimized prediction function. This can be interpreted loosely as over-fitting the models to the learning set. Cross Validation is used to limit this occurrence though the associated compute costs are significant. This research aims to explore how DP can be used instead of CV to rectify the chances of over-fitting data in a more computationally efficient way.

Though the true identity of each trained point is not of legitimate concern (actually privacy is not important), the advantages of a slight obfuscation on the learning set may yield beneficial for the overall stacking process. Computation times are decreased at the same time as limiting the risk of over-fitting too much.

4.1 The Laplace Mechanism and The Local Model

Dwork and Roth (2014) in *The Algorithmic Foundations of Differential Privacy* define a variety of mechanisms in-which random ϵ - noise may be added. Laplace noise is the simplest form of noise addition, with some optimal ϵ value dependant on the amount of privacy that is to be

afforded. Composition theorems exist for adding multiple iterations of noise, though due to the experimental nature of this topic, a closer look at these results did not prove fruitful at this time. In a purely theoretic world the ϵ chosen would not be dependant on the size of the database, but only the expected number of queries and privacy leakage deemed acceptable in each use case. In the experimental world where privacy is just a means to an end (to rectify over-fitting), this is not the case.

Noise addition can be added through two broad approaches globally or locally. The global model refers to noise added to the final results of a database query. In one example the noise added to β coefficients of a linear model trained on a set of point can be seen as a global approach. The local model relies on the composition theorem of accumulated noise over a set of observations, each data input would be slightly altered to mask its true identity from the modelling mechanism. In a linear model approach eaxample the noise added to each row of input in the X matrix would cause the $\beta = (X'X)^{-1}X'y$ coefficient to be found without viewing each points actual identity, this represents the local model in DP.

4.2 Proposed Areas of Privacy in Stacking

Within the context of Stacked Generalization, each of the generalizers used at each level is seen as a black box function. Insertion of privacy within stacking for this reason would benefit most in a local model perspective, simply being added to each of the data inputs individually. This will leave each generalizer unaffected while still (hopefully) discouraging over-fitting of models to occur.

Adding Laplace noise can be inserted in a few areas a number of times. The two algorithms here suggest inserting the noise first to the learning set such that $\Theta_j = \Theta + \xi_j$ where $\xi_j \sim Laplace(0, \epsilon_j)$. Noise is either added in the training stage of the generalizers and then a second noisy learning set is used to predict these trained values (Algorithm 5). Another approach is to use a different noisy learning set on each generalizer, then predict based on the original learning set (Algorithm 6).

Algorithm 5: Stacked Generalization with no CV + Differential Privacy $\times 2$

```
### Level 0 ###
Θ1 = Θ + Laplace(0, ε1)
Θ2 = Θ + Laplace(0, ε2)
for j in 1:k do
    Train G_j with Θ1
    Predict G_j(Θ2)
end

### Level 1 ###
Model Aggregation
```

Algorithm 6: Stacked Generalization with no CV + Differential Privacy $\times k$

```
### Level 0 ###
for j in 1:k do
    Θ_j = Θ + Laplace(0, ε)
    Train G_j with Θ_j
    Predict G_j(Θ)
end

### Level 1 ###
Model Aggregation
```

5 Experiments

Experiments were conducted to establish a deeper understanding of Stacked Generalization. The specific area of focus when conducting trial runs is the question "Does Stacked Generalization Over-fit?" In combination with a standard implementation, a variety of level 0 generalizers were tried with a large number of different linear models. At the aggregation step the (Breiman, 1996) approach was followed, using quadratic surface fitting including non-negativity, and summing to 1, constraints of the weights based on a Wolpert's original paper.

5.1 Set-up

Setting up the experiments a ground truth function of $\mu_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2$ was established with a predictor of $Y_i = \mu_i + \varepsilon_i$, such that $\varepsilon_i \sim N(0, \sigma^2)$. The goal of the experiments was to predict this Y_i as an increasing sample size (learning set) becomes available. As the experiments are completed, the hope is to discover places where over-fitting of the data could be found throughout the entire stacking process.

There are two general approaches when developing models to predict the input set; including the correct model in the generalizer set (correctly specified case) or not including it (misspecified case). The focus of these experiments was to discover over-fitting in the models where the ground truth data generating model was not represented in any level 0 generalizer, thus misspecified cases were the focus.

In the first approach, developing a set of generalizers was done with a variety of k's with hopes that increasing the number of generalizers would yield significantly over-fitted results. Experiments fit simple linear models such that $\mathcal{M}_j$ is fitted to $Y = \beta_0 + \beta_1 X^{j+2}$, where $j = 1, 2, ..., k$.

The second approach follows a set up similar to Grunwald and de Heide's discussion response to (Yuling, Y. Vehtari, A. Simpson, D. Gelman, A., 2018, 958). Experiments fit k linear models such that $\mathcal{M}_j$ is fitted to $Y = \beta_0 + \beta_1 Z_1 + ... + \beta_j Z_j$ where Z_i is equal to the $(j+2)$ degree Legendre polynomial of X and $j = 1, 2, ..., k$.

The level 1 weights in both experimental approaches were minimized with quadratic scoring, with Wolperts originally posed constraints over the variance co-variance matrix of the level 1 learning set.

$$w^* = \arg\min_w(\mathbf{wVw}) \quad w_i \geq 0, \sum w_i = 1 \tag{3}$$

$\mathbf{w}$:= weights, $\mathbf{V}$:= Variance co-variance matrix from k level 0 predictions against the true labels.

In each experiment 50 replicates were conducted, with the average Mean Square Errors (MSE) being reported. The number of level 0 generalizers $k = 10$. The sample size of the learning set increased in increments of 5 from $n = 10$ to $n = 125$. The testing set in all experiments was of size 1000.

5.2 Over-fitting

A tangent discussing how over-fitting is determined must be established before presenting any results. This is necessary as a means to discover which models are advantageous or not in the experimental process, when compared to others respectively. With no mathematically rigorous definition of over-fitting the general consensus is to compare the training and testing data sets as the appropriate means to measure the model fit.

Once all the data has been computed, the prediction error can be found. In the experiments conducted, the larger the difference in this error term between training of the defined sample size and 1000 test points generated from the ground truth model, the greater the significance of over-fitting will exists. This is plotted in an easy to interpret fashion in the following section.

5.3 Results

5.3.1 Approach 1

The initial idea of finding over-fitting was to use a large number of single linear regression models with different degree polynomials being fitted. When aggregating over the models it was noticed that only a small number of weights at level 1 were non-zero. These results are similar to (Breiman, 1996) findings. The non-zero weights tended to favour the lower order polynomial fits, though there was no easily interpreted pattern in these findings.

When comparing figures 1 and 2 we can see the both of the MSE's are quite close to the optimal results possible (defined by the red line). As each function is so close to representing the true deviance of the data's noise, when comparing LOO CV and no CV each figure shows little over-fitting has occurred. One hypothesis that explains this is due to the simplicity of the models used across level 0 generalizers. No further experimentation to rectify the over-fitting was conducted as these results would be to difficult to establish success given such a small amount of prediction differences between the learning and testing sets in both experiments.

5.3.2 Approach 2

When developing a set of level 0 generalizers that was more complex, it appears as though a significant amount of difference has occurred (comparing figures 1 and 3). Computation times were also increased as growing "multiple linear regression" models needed to be computed. It seems to take more time for the stacking model to converge closely with the learning set and test set (as shown

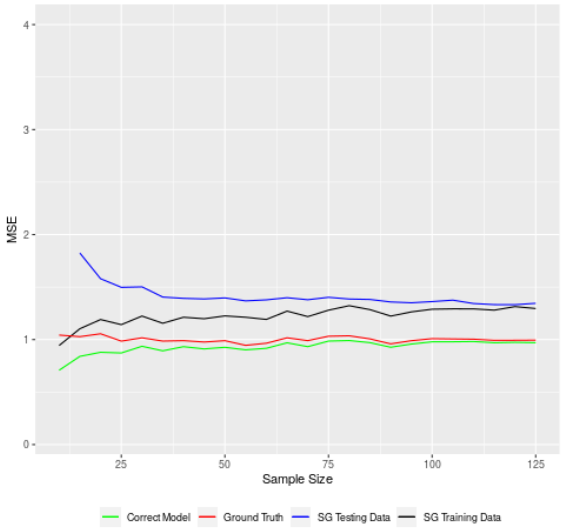

Figure 1: With LOO CV (Approach 1)

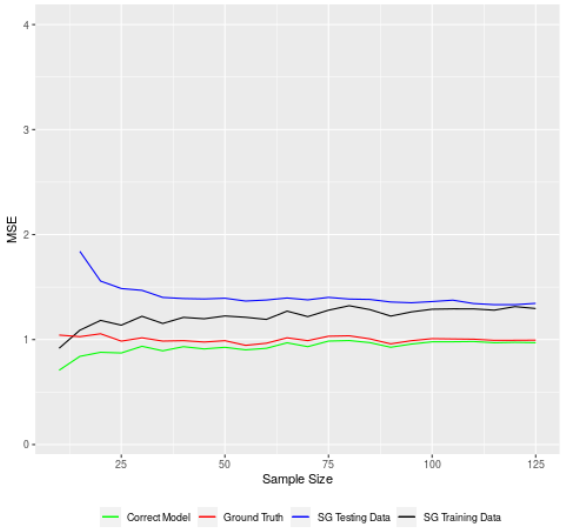

Figure 2: With-out CV (Approach 1)

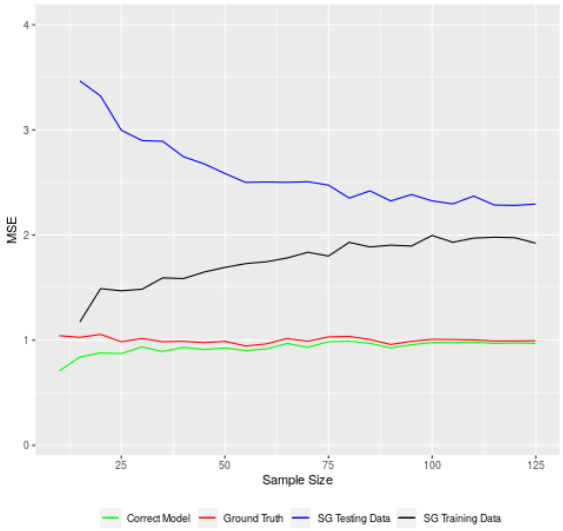

Figure 3: With LOO CV (Approach 2)

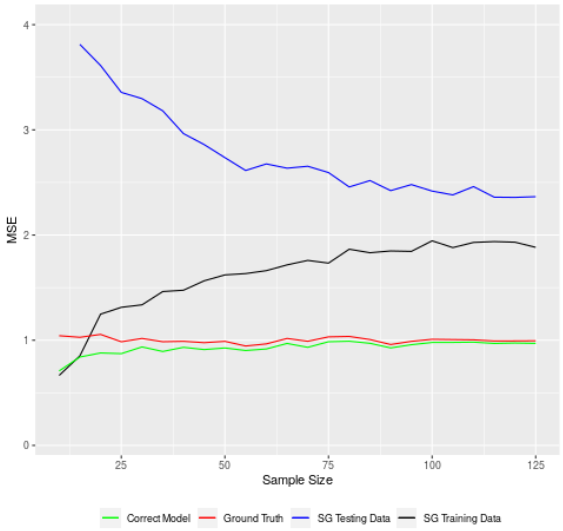

Figure 4: With-out CV (Approach 2)

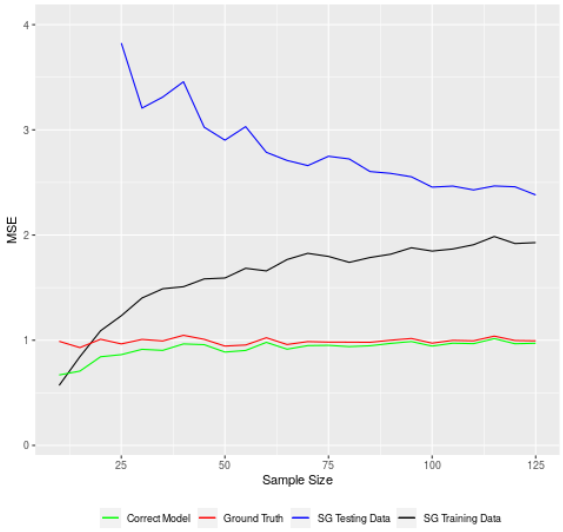

Figure 5: With-out CV (Approach 2)
With Differential Privacy × 2

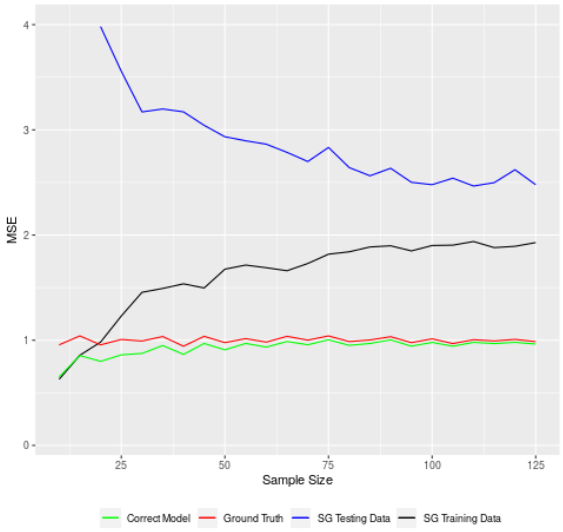

Figure 6: With-out CV (Approach 2)
With Differential Privacy × k

in figures 3 and 4). This is a prime indicator of over-fitting. Compared to the previous approach, the prediction results do not seem as accurate, though a variety of reasons may be the cause for this.

When initially implemented Legendre polynomials were not used, creating an ill-conditioned matrix that produced useless results. Once these Legendre polynomials were included (in the model figures used), the results became much more realistic. A benefit of this approach is that the linear model coefficients of each generalizer learned where not likely to produce co-linearity across different polynomial factors.

Interestingly, there were much fewer weights with a value of zero across experiments; unlike in the previous approach with the simpler linear models. These weights also seemed to concentration around medium sized model complexities, potentially representing some form of regularization effect. No clear pattern was analysed, just elementary observations.

With the potential of over-fitting found, the experiments in figures 5 and 6 were conducted by applying differential privacy according to algorithms 5 and 6. Here ϵ values were used as a small fraction of the set error in the generating function (σ^2). It does not appear from these results that differential privacy helps in eliminating any over-fitting, in-fact these experiments almost show an increased amount of it. Results were not tuned for the optimal ϵ value, used in preserving privacy.

6 Future Work

Searching for and addressing over-fitting, resulted in the most significant struggle throughout the course of this research. Finding ways where stacking over-fits is the most important issue that needs to be explored in greater depth. Though an indicator of stacking was found in the experiments, it may prove to be more useful to increase this to a larger number of experiments utilizing significantly different generalizers than traditional linear models. Discovering ways to mitigate these situations

is the next most important stage that can be addressed, with some suggestions in the Differential Privacy section.

6.1 Experimental Design

- Classification Data

  Experiments were only completed on regression data. It would be interesting to see how more classification focused algorithms such as Decision trees and K-means would work in a stacking environment.

  Level 1 aggregation models utilizing squared error losses may be less effective with discrete data, though further experimentation would be necessary with a variety of different non-linear objective functions being minimized.

- Higher Dimensional Inputs

  The experiments were completed with on a single dimension (d) of input, fixed where $x \in [-1, 1]^d$. Increasing this to a higher space would increase the sparsity of data samples, causing more over-fitting to occur.

- Higher Dimensional Outputs

  Predictions were only made with a single regression value as output. As output variables increases, different ways in measuring a predictions accuracy for every generalizer can be used, as one dimension of output may be more accurate than others. The objective functions that would be minimized have the potential of being more complex, with aggregation weights also having the potential of greater sophistication. One such example, each dimension could be aggregated separately if predicted values are assumed independent.

6.2 Level 0

Discussed in the experimental design section, increasing the dimension of the inputs has the potential to greatly increase the potential of creating sparse data which is more prone to over-fitting. This section focuses more on the way the data is transformed and enters into and is used in the generalization stage of stacking.

- Inputs

  In the second experiment orthogonal Legendre polynomials were utilized to transform a single x value into a higher number of orthogonal inputs that represent differing polynomial degrees (in experiments from degree 3 to $(k + 2)$). Other basis functions can be used, with differing amounts of collinearity. Generally complete independence is necessary for a generalizer to work optimally, though this is case dependent.

  Another interesting approach would be to instead use trigonometric basis functions to transform the input data. The hope is to create a more usable transform specific to the generalizer being used. A combination of these transformations may also yield interesting results when used on a similar modelling technique (using the same learning algorithm with different transforms on the same data as separate level 0 generalizers).

- Generalizers

  All of the generalizers looked at in these experiments were based strictly on linear models. The reason for using this model is based on its well understood and interpretable properties.

Once a better understanding of how over-fitting may be produced, novel results may be found using a variety of models with different methodologies mixed together.

6.3 Level 1 & Beyond

- Non-negativity Constraints

  Non-negativity constraints are a helpful way to guarantee the level 1 generalizer is not attempting to over compensate for discrepancies in the level 1 learning set. Setting a model weight to have negative importance creates a model with a more difficult to interpret output. Stripping this constraint when minimizing the level 1 objective function, could produce a model with increased precision, and increase the potential for over-fitting to a specified set of data. These concepts were explored in-depth in (Ting and Witten, 1999). Their conclusion was that the non-negativity constraints are necessary for regression problems though not for classification problems. A further exploration through experimentation could produce interesting results and intuition behind this situation.

  One specific question that arouse from the first approach of experiments conducted, was if certain generalizers had weighting that was correlated with on another (similar models being negative while others positive, etc.).

- Stacking of Objective Functions

  The majority of papers only discuss the potential of stacking at the zeroth and first level. (Wolpert, 1992) developed stacking with the idea that this process can be conducted iteratively until some desired level has been reached. The issue with increasing levels is finding ways to improve the predicted results are scarce. One suggestion is to use a collection of objective functions all minimizing the aggregation weights of the level 0 generalizers differently. These different generalizers would then be used to create a level 2 learning set and so forth.

6.4 Differential Privacy

- Concerns

  A risk associated with these noise additions of DP is the unaddressed question "Does this avoid over-fitting or just cause over-fitting of a separate dataset?" Further exploration of these concepts must be addressed to better understand how privacy can be applied.

- Homoskedastic Noise

  The majority focus of this research evolved around producing over-fitting. Time spent tuning of ϵ in differential privacy is an easy way to improve prediction results. Searching for algorithms to discover the optimal value to add noise while minimizing the privacy lent could help to rectify the issues created by over-fitted models.

- Heteroskedastic Noise

  In practical applications choosing ϵ will be dependant on the data. Approaching data insertion can then be done where each noise element $\xi_i \sim Laplace(0, \epsilon_i)$ is added to individual points x_i for all $i = 1, 2, ..., n$. Each ϵ_i is determined based on the influence that each data point has on the overall model. This approach may help to mask the points importance when queried many times by different generalizers in the training process.

7 Conclusion

The conclusion to this research is promising. Starting out, a search for over-fitting in stacked generalization has proven to be harder than originally anticipated. After increasing the model complexities of level 0 generalizers, signs of an over-reliance on the training data have begun to show. Rectifying these issues at this stage is a large venture, as the intuitive reason for over-fitting in this process has not clearly been established. Once LOO CV has been removed, while significantly decreasing computing costs, it is easy to observe overly trained generalizers. With this adapted version of stacking, the potential of fixing issues caused with the lack of CV may be assisted with new techniques in DP, though again a further exploration of the initial problem is necessary.

With a spectrum of different linear models being applied with a changing number of generalizers, stacking clearly show to perform better than any one of these models, though the optimal model combination techniques are still not clear. Testing the importance of assumptions and changing the minimizing objective function used shows to be a place where improvements can be made when moving to a higher level in the stacking process.

Moving forwards a large body of work still remains to be completed with section 6 highlighting the starting points in different areas.

8 References

[1] Wolpert, D.H. (1992). Stacked Generalization. *Neural Networks*, 5(2): 241-259.

[2] Breiman, L. (1996). Stacked Regressions. *Machine Learning*, 24(1): 49-64.

[3] Ting, K. M. and Witten, I. H. (1999). Issues in Stacked Generalization. *Journal of Artificial Intelligence Research*, 10: 271-289.

[4] Clarke, B. (2003). Comparing Bayes Model Averaging and Stacking When Model Approximation Error Cannot be Ignored. *Journal of Machine Learning Research*, 4: 683-712

[5] Van Erven, T. Grünwald, P. and de Rooij, S. (2012). Catching up faster by switching sooner: a predictive approach to adaptive estimation with an application to the AIC-BIC dilemma. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(3): 361417.

[6] Yao, Y. Vehtari, A. Simpson, D. Gelman, A. et al. (2018). Using stacking to average bayesian predictive distributions. *Bayesian Analysis*. 13(3): 917-1003.

[7] Grünwald, P. and Van Ommen, T. (2017). Inconsistency of Bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4): 1069-1103.

[8] Dwork, C. Roth, A. (2014). The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4): 211-407.