”How does sample weighting improve learning curve fitting?”

Lapo den Hollander
Supervisor(s): Tom Viering, Taylan Turan, Cheng Yan

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
January 26, 2025

Name of the student: Lapo den Hollander
Final project course: CSE3000 Research Project
Thesis committee: Tom Viering, Taylan Turan, Cheng Yan <Examiner>

Abstract

Learning curves plot the performance of a machine learning model against the size of the dataset used for training. Curve fitting is a process that attempts to optimize algorithm parameters by minimizing the error in its loss function, thereby achieving the best possible fit to the data. We apply various sample weighting techniques to the curve fitting process and evaluate whether the resulting weighted curves can significantly improve the performance of the model. We explore whether adjusting the magnitudes of these weights can further improve the fit of the curve. The results demonstrate that each sample weighting method, as well as larger weight magnitudes, can significantly improve error rate prediction for anchors beyond the range of the observed data.

1 Introduction

An important aspect of the development of a machine learning model is its ability to improve its performance in various tasks, such as classification, regression and clustering. The performance of a machine learning model is heavily dependent on the number of data samples used when training said model[1]. Understanding the relationship between the amount of data available to train a model and its performance is essential, particularly for optimizing training data allocation and improving the accuracy of test set error rate prediction. The demand for efficient machine learning models continues to grow, further emphasizing the importance of understanding and optimizing learning curves to develop models that balance accuracy and computational cost.

Learning curves are functions that plot the amount of data used to train a machine learning model against its performance when labeling the data reserved for testing it. In other words, they offer a visual representation of how a model's performance changes as the amount of available training data increases. In order to obtain learning curves from these measurements, the process of curve fitting is applied, which involves optimizing the parameters of a chosen parametric function to closely approximate the relationship between training data size and model performance.

Sample weighting plays an important role in data analysis by addressing imbalances within a dataset, allowing for a more equal representation of all data points. For example, sample weighting can address class frequency imbalances by assigning higher weights to sparse classes, allowing these points to have more influence during machine learning model training and preventing bias towards the more prevalent class. Sample weighting is common in supervised machine learning practices such as classification tasks [2] and latent variable modeling [3]. However, this practice remains relatively unexplored within the field of learning curve fitting. This paper aims to answer the following question: what is the effect of a number of sample weighting methods on the learning curve fitting process? The goal is to discover whether these approaches can significantly improve the accuracy of fit.

2 Related Work

This paper builds upon various previous relevant academic works. Predominantly, a review on learning curve shapes by Viering & Loog (2023) remarks that a great number of studies that compare various parametric models with the goal of finding the overall best one for a set of learners reach contradicting conclusions when compared to each other [4]. They mention some of the most common pitfalls they have observed such as only fitting to linear curves, not including a bias term in the parametric model, not performing extrapolation to unseen data and no metric to measure statistical significance of the results. They also mention that extrapolation should be used in addition to interpolation for more accurate measurements [4]. Mohr *et al.* (2023) also provide the LCDB database [5], an extensive database containing learning curves trained on a large number of datasets, which we use for our learning curve analysis.

Furthermore, one of the most extensive learning curve studies was performed by Brumen *et al.* (2014), who attempted to investigate which curve function mathematically best fits the C4.5 algorithm, which can generate decision trees. They compare a power, linear, logarithmic and exponential function [6] on a large number of datasets. They find the exponential function to outperform the other three in 64 out of 86 cases on well-behaved curves. Well-behaved learning curves show decreasing error rates as training set size increases. However, they only analyze a single learner and they occasionally reduce the number of samples selected from the anchor data in order to keep this data from being ill-behaved, thus potentially masking inferior results if the entire dataset was used. We aim to investigate multiple learners and utilize the full range of data for each dataset used for analysis.

Additionally, Figureoa *et al.* (2012) analyzed curve fitting with sample weights by assigning larger weights to points with a larger index. More specifically, each point is assigned a weight equal to i/j, where i is the index of the point and j is the anchor of the dataset containing it. They found that in nearly all cases, their weighted algorithm outperformed the non-weighted alternative[7]. Although the experiments performed with these sample weights are plentiful, there are a number of problems. They randomly split the data for fitting and evaluating, instead of extrapolating it. This limits the confidence with which can be claimed that the results show improved error rate prediction for larger anchors. Additionally, they perform a paired t-test, which assumes the differences between paired samples are normally distributed. This assumption is not supported by any evidence, however, questioning the robustness of their conclusions. The paper also has some limitations. It only includes a single sample weighting method and a single set of weights, thus limiting the broadness covered by the results. This paper aims to address these limitations by exploring various sample weighting methods, adjusting their magnitudes to measure their impact, performing extrapolation alongside interpolation and measuring significance with Wilcoxon signed rank test.

Finally, many meta-analysis papers perform inverse-variance weighting on the data based on various precision metrics, such as sample size variability between different studies [8] or measurement error [9]. Assigning weights based on the variance allows these studies to ensure that more precise data points exert more influence on the analysis. Specifically, Marín-Martínez and Sanchez-Meca (2010) demonstrate that the estimator found by Hedges and Vevea (1998), which uses inverse variance weighting, performs better in terms of mean-squared error compared to alternative methods in their meta-analysis [10]. However, despite its widespread use in meta-analysis, inverse-variance weighting has not yet been widely studied in the field of learning curve fitting. Our aim is to investigate whether inverse-variance weighting could provide utility in our research when applying sample weights to curve fitting on the mean of multiple curve splits by using the standard deviation of the set of corresponding points across those splits.

3 Background

This section provides formal definitions and mathematical explanations for learning curve fitting, learning curve evaluation and sample weighting.

3.1 Learning Curve Fitting

Learning curves map the size of the training dataset used on a machine learning model to the error rate of their test data label prediction after training, as defined by Mohr & van Rijn (2022) [11]. They can take various forms, but many have the shape of an exponential or power curve. Figure 1 shows an example of a learning curve with an exponential shape. Here, the training dataset size (anchor) is plotted on the x-axis and the error rate of the test data label prediction (error rate) on the y-axis.

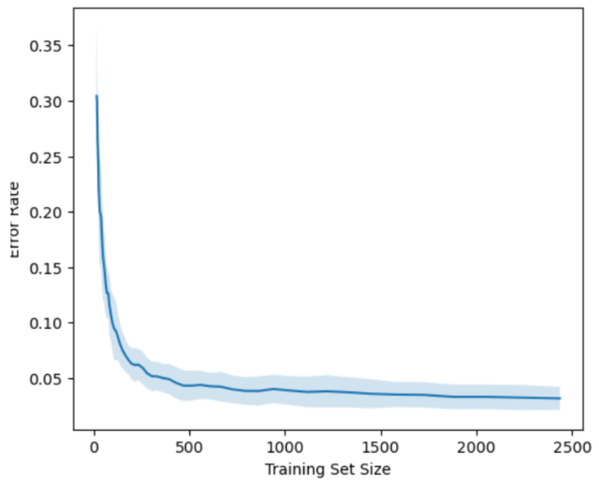


Figure 1: Example of a learning curve

Curve fitting is the process of finding some mathematical function that accurately represents the relationship between a set of input data points and its corresponding output values. In the context of learning curves, this involves selecting a parametric function, such as an exponential or power-law function, and optimizing its parameters to minimize the difference between the observed error rates and the error rates predicted by the curve for the same anchors. This error is typically quantified using a loss function, such as mean squared error (MSE). MSE measures the average squared differences between predicted and observed values. Let Y denote the set of observed output values for the set of input values, $\hat{Y}$ the set of predicted output values for the set of input values and n the total number of data points in the dataset. The MSE is defined as follows:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Let X denote the set of input values, θ the set of parameters of the model that should be optimized and $f(X, \theta)$ the curve function that predicts the output for the given input and set of parameters. The following formula describes curve fitting with mean-squared error as the loss function, where we attempt to minimize its outcome:

$$\min_{\theta}\sum_{i=1}^{n}\left(f(X_i;\theta) - Y_i\right)^2$$

3.2 Learning Curve Evaluation

After data is used to fit a curve to a set of points, its accuracy must be evaluated by measuring how well the fitted curve or model predicts output variables given a set of unseen data points. In order to do this, we exclude a subset of the data from fitting and reserve it for evaluation, so as to not over-fit to the fitting data. Common approaches for splitting in this manner are extrapolation and interpolation. "Interpolation is a method to find the value between the known data points" [12]. It helps us answer questions about the immediate data and allows us to evaluate data of specific sizes that have already been observed. Mathematically, this involves splitting the data such that the evaluation points, denoted as $\mathbf{x}_{\text{eval}}$, fall within the range of the fitting data points $\mathbf{x}_{\text{fit}}$ without overlap.

$$\mathbf{x}_{\text{eval}} \in (\min(\mathbf{x}_{\text{fit}}), \max(\mathbf{x}_{\text{fit}})) \quad \text{and} \quad x_{\text{eval}} \cap x_{\text{fit}} = \emptyset$$

.

Interpolation alone provides limited insight into how the curve performs on data outside the observed range. "Extrapolation is a method that estimates a value beyond the range of a given set of data" [12]. It aids in predicting whether the performance of a model continues to improve with increasing (or decreasing) data, or starts to plateau. Extrapolation is particularly useful in the context of learning curves, as performance tends to become a greater concern when datasets grow larger. Extrapolation is, in fact, a popular evaluation technique within the medical field as well, as exemplified by Mukherjee et al., (2021), where dataset size requirements for DNA microarray data classification were extrapolated [13]. In mathematical terms, extrapolation involves splitting the data such that the evaluation points, $\mathbf{x}_{\text{eval}}$, fall outside the range of the fitting data points $\mathbf{x}_{\text{fit}}$. This means the evaluation points follow these rules:

$$\mathbf{x}_{\text{eval}} < \min(\mathbf{x}_{\text{fit}}) \quad \text{or} \quad \mathbf{x}_{\text{eval}} > \max(\mathbf{x}_{\text{fit}})$$

.
A visual example of how the fitting and evaluation data might be divided is shown in figure 2.

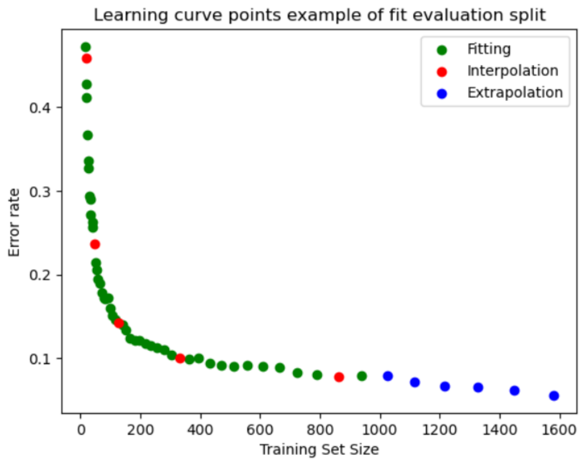


Figure 2: Example of fitting and evaluation data division.

3.3 Sample Weighting

Sample weights skew the importance of individual data points in proportion to others in model training by assigning different weights to each point. This is achieved by multiplying each data point's contribution to the loss function by its corresponding weight, adjusting their relative influence on the model's optimization process based on the weighted sum. Let N denote the total number of data points in the dataset, $X = \{x_1, \ldots, x_n\}$ the input data points, $Y = \{y_1, \ldots, y_n\}$ the output data points, $W = \{w_1, \ldots, w_n\}$ the weights, $\ell(Y_i, f(X_i))$ the loss function, which measures the difference between the observed output value and the predicted output value for a given input value X, $f(X)$ is the model's prediction for the input X and L the sum of every product of the weights and the loss for all data points in the dataset. The mathematical notation is as follows:

$$L = \sum_{i=1}^{N} w_i \cdot \ell(Y_i, f(X_i))$$

4 Methodology

This section outlines the methodological framework of the paper, covering the parametric model, data splitting for fitting and evaluation, curve fitting inputs, and the resulting outputs.

4.1 Parametric Model

We use an exponential curve with three terms for learning curve fitting. This curve is equivalent to the EXP3 function defined in the LCDB paper [5] by Mohr *et al.* (2023), found to be effective by Brumen *et al.* (2014) for learning curve fitting on well-behaved curves, as mentioned previously [6]. The terms of this exponential function consist of the scaling factor a, which determines the function value when $x = 0$. The decay constant b controls how quickly the function grows or shrinks. Finally, the bias term c is the asymptotic value that the function approaches as x becomes very large.

$$\text{EXP3: } y = a \cdot e^{-b \cdot x} + c$$

4.2 Data Splitting: Fitting and Evaluation

To evaluate the curves, we split the dataset $D = \{x_1, \ldots, x_n\}$ into a fitting set F and an evaluation set E, such that $F \cup E = D$ and $F \cap E = \emptyset$. Let n_{fit} and n_{eval} represent the sizes of F and E, respectively, with $n_{\text{eval}} = n - n_{\text{fit}}$.

The evaluation set E is further divided into two equal parts: the extrapolation set E_{extrap} and the interpolation set E_{interp}. The extrapolation set consists of the last k points of D, while the interpolation set is selected evenly from the first $n - k$ points of D, where $k = n_{\text{eval}}/2$. The sets are defined as follows:

$$E_{\text{extrap}} = \{x_{n-k+1}, \ldots, x_n\},$$
$$E_{\text{interp}} = \{x_{j \cdot s} \mid j = 1, \ldots, k,\ j \cdot s < n_{\text{fit}}\},$$

where $s = n_{\text{fit}}/k$ is the step size for interpolation.

The fitting set F contains all points not included in E.

4.3 Curve Fitting Inputs

Based on a suggestion from D. Darie, a collaborator on this research project and author of a related study (D.Darie, personal communication, January 09, 2025), we utilized the Levenberg-Marquardt (LM) algorithm for curve fitting. LM is a nonlinear least-squares optimization technique used for solving regression problems. It dynamically adjusts a control parameter λ to interpolate between the stability of gradient descent and the speed of the Gauss-Newton method, balancing efficiency and robustness in curve fitting applications [14]. Furthermore, each anchor point represents the mean of corresponding points across every training, validation, and test split. The validation set is defined as "the set of examples used for model selection" by Tennenholtz *et al.* (2018), who used the validation set after training various models for model selection [15].

As detailed earlier, sample weights influence a data point's contribution to the loss function, so we apply them during curve fitting. We explore three different sample weighting methods. These include sample weighting based on the anchor, the error rate, and the standard deviation of the error rate on the mean curve (SD-Error). This parameter represents the standard deviation of the corresponding points across multiple splits of that dataset, effectively using the variability of each point's value across these splits as its weight. Define $X = \{x_1, x_2, ..., x_n\}$ as the set of anchors, $Y = \{y_1, y_2, ..., y_n\}$ the set of error rates, $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_n\}$ the set of SD-Error values, $W = \{w_1, w_2, ..., w_n\}$ the set of weights and s the scale of the weight. The scale s adjusts the influence of each data point by applying an exponentiation to the values, raising each weight to the power of s. A positive scale assigns weights to a dataset that increase with their value, a negative scale assigns decreasing weights instead. The weights are obtained as follows:

$$w_i = \frac{x_i^s}{\sum_{j=1}^{N} x_j^s}$$

$$w_i = \frac{y_i^s}{\sum_{j=1}^{N} y_j^s}$$

$$w_i = \frac{\sigma_i^s}{\sum_{j=1}^{N} \sigma_j^s}$$

4.4 Curve Fitting Outputs

The effectiveness of each sample weighting method and weight scale combination is measured by the average MSE over all datasets for interpolation and extrapolation. Subsequently, a significance test over the MSE values obtained by evaluating a weighted curve and those from a non-weighted curve is performed. Both sets are paired, as the same model has been fitted to the same dataset but with different weighting strategies, and not normally distributed. We use the Wilcoxon signed-rank test, a non-parametric rank test, frequently utilized for comparison between two paired sets of data, exemplified by Neumann *et al.* (2003) where the Wilcoxon signed rank test was used to assess the significance of improvements in endoscopic performance parameters during a training course [16]. The null hypothesis H_0 states: there is no significant difference in the MSE between curve fitting with uniform weighting and curve fitting with sample weighting at the specified weight scales.

5 Experimental Setup

5.1 Data Collection

We use the learners and datasets from the LCDB database [5], which was provided to us by our supervisors. A total of 24 learners were trained on 72 datasets from the OpenML database [17]. In other words, each learner comes with a premodeled set of mean anchor data over various train, validation and test splits. As mentioned previously, we prioritize more monotonic datasets as the exponential function performs best on monotonic and convex data. We were provided with a table that indicates how monotonic and convex each learner is (C. Yan, personal communication, December 05, 2024). This table can be found in Appendix A. We hypothesize it to be important to use as many datasets as possible to get a good representation of the behaviour of a learner. Due to processing constraints, we therefore decide to use every dataset but limit the set of learners used to five only. These learners are shown in table 1.

Learner	**Monotonic & Convex**
ens.GradientBoosting	100.00%
Decision Trees	97.22%
SGDClassifier	97.22%
ExtraTrees	95.83%
Perceptron	95.83%

Table 1: Monotonicity and convexity percentage of the anchor data points for each of the learners used

5.2 Pipeline

We split our data into a fitting set and an evaluation set using an 80/20 split, meaning 80% for fitting and 20% for evaluation. This has been found to be a relatively computationally efficient, as well as considerably stable approach to splitting data by Brumen *et al.* (2021) [18]. The final 10% of the dataset is used for extrapolation. We hypothesize this extrapolation set to perform very effectively for sample weighting methods that emphasize points toward the end of the learning curve, as such we reserve another 10% of the dataset, between the first 90% of the dataset, for interpolation. The remaining data is used for fitting. This approach is common, as exemplified by the work of Kolachine *et al.*, who apply extrapolation on inferred learning curve data for error rate prediction on unseen data [19]. The function we use for curve fitting is exclusively the EXP3 function mentioned before.

As mentioned previously, each learner in the LCDB database was trained on the training split of the corresponding dataset. However, it was evaluated on every split separately. Preferably, we want to measure the performance of a learner by measuring how well it fits to unseen data, as is typical, namely the validation or test set. Measuring performance on the same data that was used for training likely causes overfitting. Due to the high similarity between the mean curves of the learner-dataset combinations for validation and test sets, it is redundant to evaluate both. We deem both to be equally suited, therefore, we focus solely on the test set.

We generate three sets of sample weights, each based on specific data-driven characteristics. We generate a set of weights based on the anchors (x), another based on the error rates (y) and finally one based on the SD-Error (σ) of the mean curve. The weights are always equal to the values before normalizing; for example, if the anchors are equal to $\{1, 2, 3\}$, the weights equal $\{1/6, 2/6, 3/6\}$.

We generate a set of weight scales, raising each set of weights to the power of a scale in order to influence the magnitude of the weights. We define the magnitude of the weight to be the absolute value of the weight. Due to time and processing constraints, we resort to a set of 14 manually selected weight scales S, as seen in table 2.

N.S.	-4.0	-3.0	-2.0	-1.0	-0.75	-0.5	-0.25
P.S.	4.0	3.0	2.0	1.0	0.75	0.5	0.25

Table 2: Table of positive and negative weight scales

S is selected such that we evaluate weights that both increase and decrease with the original data at both amplified and reduced magnitudes. For clarification, let us assume a set of weights $W = \{1, 2, 3\}$ and a scale s. If $s = 0.0$, then we raise each weight to the power of 0. This gives us $W_s = \{1, 1, 1\}$. This is equal to uniform weights and thus redundant, so we do not include it in S. If $s = 1.0$, the weights remain the same and if $s = -1.0$, $W_s = \{1, 1/2, 1/3\}$. If $s > 1.0$, we increase the magnitude of the original values on the weights, and if $0.0 < s < 1.0$ we decrease it. This also goes for negative weights, the difference being that negative weights decrease when the original data increases.

We chose scales between -4.0 and 4.0, again due to processing limitations. We hypothesize that this will have minimal impact on the conclusions, as the focus of this paper is on determining whether weight scaling can enhance sample weighting in general, rather than identifying an optimal scale value per sample weighting method.

We randomly generate the three terms for the exponential function from a uniform distribution, as it was found by D. Darie that uniformly generating parameters is typically superior to generating them based on a normal distribution on well-behaved curves, which include monotonic curves (D.Darie, personal communication, January 09, 2025). We run the `curve_fit` method from the scipy optimize library [20] with these terms, the exponential function using the LM algorithm. We generate 100 curves by running `curve_fit` 100 times, measuring the MSE on the evaluation points each time, which consist of both interpolation and extrapolation points. Finally, we pick the curve that achieves the lowest combined MSE across the interpolation and extrapolation evaluation points. This allows us to prevent outliers influencing the results. We generate 43 curves total. One without sample weighting (uniform weights) and then 14 for each of the sample weighting methods mentioned above at every scale.

6 Results

We present the MSE values for the Gradient Boosting learner curve fitting with uniform weights and scaled sample weights in detail by plotting them in a box plot, highlighting their mean, median and variance. Figure 3 shows the MSE values obtained through interpolation, figure 4 the MSE values obtained through extrapolation. These box plots clearly show that each sample weighting method performs at least marginally worse than uniform weighting at even the optimal scale for interpolation. Meanwhile, they clearly outperform uniform weighting for extrapolation, showing smaller means and medians, and significantly smaller variances.

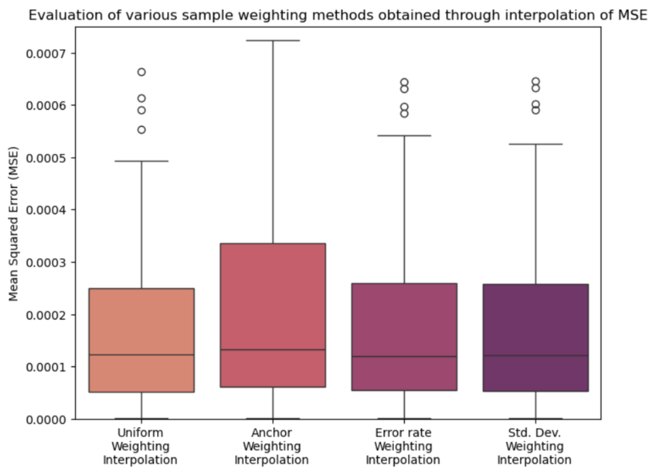


Figure 3: Boxplot comparing Gradient Boosting learner MSE values for uniform weights and scaled sample weights obtained through interpolation.

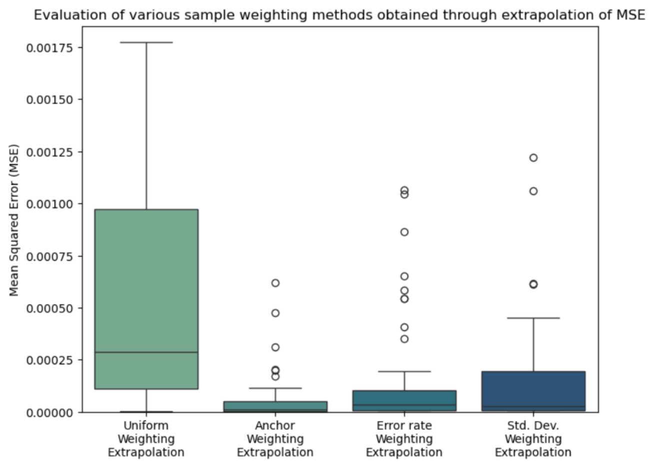


Figure 4: Boxplot comparing Gradient Boosting learner MSE values for uniform weights and scaled sample weights obtained through extrapolation.

We then present the results more precisely by giving two tables, one for interpolation and one for extrapolation, that contains relevant data for each sample weighting method. More specifically, we show the mean MSE over all curve MSE values of the sample weighting method and the optimal weight scale. Table 3 shows the interpolation results, table 4 the extrapolation results. From the results we can see that none of the sample weights performed better than the uniform weights when evaluated through interpolation. The optimal scales indicate that on these datasets, curve fitting with sample weights that are decreasing on anchor, increasing on error rate and increasing on SD-Error performs best, with relatively low magnitudes being better. Evaluation through extrapolation performed much better, however. Every sample weighting method outperforms uniform weighting. The optimal scales indicate that sample weights increasing on anchor, decreasing on error rate and decreasing on σ outperform uniform weights, with relatively high magnitude weights being better.

Method (Interpolation)	**MSE Mean**	**Optimal Scale**
Uniform Weighting	1.94e-04	0.0
Anchor Weighting	2.35e-04	-0.25
Error Rate Weighting	1.98e-04	0.25
SD-Error Weighting	1.98e-04	0.25

Table 3: Comparison of Gradient Boosting learner learning curve fit MSE means and optimal scales across sample weighting methods obtained through interpolation

Finally, we present the P-values obtained from performing Wilcoxon signed rank test on the curve fits obtained with uniform weights and one of the scaled sample weights, along with whether the difference is significant. We reject the null hypothesis for $P < .05$. Because no sample weighting method outperformed uniform weighting for interpolation, we only present the results for the Wilcoxon signed rank test for ex-

Method (Extrapolation)	MSE Mean	Optimal Scale
Uniform Weighting	7.23e-04	0.0
Anchor Weighting	4.89e-05	4.0
Error Rate Weighting	1.26e-04	-4.0
SD-Error Weighting	2.18e-04	-4.0

Table 4: Comparison of Gradient Boosting learner learning curve fit MSE means and optimal scales across sameple weighting methods obtained through extrapolation.

trapolation in table 5.

Method (Extrapolation)	P-value	Is Significant
Anchor Weighting	$<.001$	Yes
Error Rate Weighting	$<.001$	Yes
SD-Error Weighting	$<.001$	Yes

Table 5: Comparison of Gradient Boosting learner P-values and significance across methods evaluated with extrapolation (excluding Uniform Weighting).

The results for the other learners are only summarized, their boxplots can be found in appendix B to get an overview of the MSE values. For each learner and sample weighting method, the significance test resulted in a P-value of $P<.001$, as such a table of these results would be redundant. Every learner outperformed uniform weighting with each sample weighting method when evaluated using extrapolation. None of the learners did so when evaluated using interpolation. Every learner showed optimal performance with weights increasing on anchor, decreasing on error rate and decreasing on SD-Error. SGD classifier and Perceptron showed interesting results, finding smaller optimal weight magnitudes for extrapolation, notably Perceptron with a weight scale of -0.33 for sample weighting based on SD-Error.

7 Discussion

These results show that sample weighting on high magnitude weights increasing on anchor, decreasing on error rate and decreasing on SD-Error can result in significantly more accurate estimation of error rate at higher, future unseen anchor values.

The results show a vast and clear difference between evaluation on extrapolation and interpolation. None of the five learners performed better with any of the sample weighting methods at any weight scale than with uniform weighting for interpolation. This indicates that sample weighting does not improve the performance of a learning curve for predicting error rate for anchor values within the observed data. Conversely, as we hypothesized, every learner performed significantly better with every sample weighting method than uniform weighting for extrapolation. Lower MSE means and reduced variances were found, particularly at weights with high magnitudes. Notably, sample weights increasing on anchors and sample weights decreasing on error rates or SD-Errors performed quite well, especially when using large weight scales. These results highlight the potential of sample weighting for providing improved predictions about error rate for anchors beyond the observed range, although these do compromise interpolation performance.

However, these findings also come with limitations. First, the improved performance in extrapolation likely arises due to the ability of sample weights to allow for skewing the focus of the curve fitting process towards specific regions of the data. It logically follows that, given that we evaluate an entire dataset only on its last points, any technique that shifts the focus of curve fitting towards points that occur toward the end will perform better. Important to note is that we are only evaluating on well-behaved curves, where we have the guarantee that no sudden peaks might appear, and the values tend to flatten. This works very well with the obtained sample weights, and thus suggests that we cannot apply the same conclusions to ill-behaved curves, which might present large fluctuations that cannot be addressed with the current approach.

Additionally, the experimental setup was limited by time and processing constraints, as well as the high computational demands of repeating the curve fitting process over many iterations to minimize the risk of outliers. These limited the scope of certain parts of our analysis, the first of which caused us to use a single 80/20 split for fitting and evaluation. Variability across different splits could have been accounted for by performing cross validation, and could allow for more robustness against over-fitting.

Finally, the time constraints and hardware limitations lead us to rely on a set of hard-coded weight scales instead of a more adaptive approach to optimal weight scale calculation, which further limits the scope of the analysis. Moreover, the use of only five learners may reduce the robustness of the findings, as the limited number of learners and the low diversity between them can limits the conclusions we can draw about general learning curve behaviors.

8 Conclusion

8.1 Summary

In conclusion, this paper investigated whether applying various sample weights during the learning curve fitting process provides significant improvements in performance over uniform weighting. It compared various curves that were generated using weights based on the anchor, error rate and SD-Error of the learning curve data. These methods were evaluated based on their MSE, and promising curves were further analyzed for significance with Wilcoxon signed rank test. The results were provided in tables and plots, showing how curve fitting with sample weights that increase with the anchor, decrease with the error rate and decrease with the SD-Error can show significant improvements compared to uniform weights in predicting error rate for anchors beyond the observed range of the data. However, the small number of learners, the use of a single fitting evaluation split and reliance on hard-coded weight scales, caused by processing power and time constraints, limit the scope of the research; the absence of these limits could enhance the robustness of the findings and make them more generalized.

8.2 Future Improvements

Future research could expand on this study by evaluating a broader range of learner dataset combinations, including ill-behaved ones, in order to give more generalized conclusions. Additionally, exploring dynamic or continuous methods for selecting optimal weight scales could improve their applicability in practice. An example of such a method would be exploring a large set of weight scales with a more accurately defined range, fitting a curve to the results and differentiating it to find the smallest scale. Moreover, performing multiple data splits, which would provide more robust evaluations for train, validation and test splits, could make the results more robust. Finally, access to more processing power through large servers could expand the scope of the research analysis, enabling the exploration of more complex models and larger datasets.

9 Responsible Research

Our research has been conducted ethically and transparently. The privacy of the data sources used have been integral throughout the entirety of the project. Although our study does not involve sensitive personal data or human subjects, it still demands that any practices in the handling of the data be done carefully and responsibly while also remaining sufficiently reproducible.

An important consideration we made was the potential introduction of bias in the choice of sample weighting methods used. If not considered adequately, these choices could lead to results and conclusions about the performance and applicability of machine learning models that can be misleading. To mitigate this, we considered the most common and contextually logical sample weighting methods appropriate for learning curve fitting and thoroughly evaluated them on their applicability and relevance. Using them across many different datasets and learners in the LCDB dataset, we analyzed their potential for improving the performances of various machine learning models. The data sets and learners provided to us by our supervisors have been carefully evaluated to ensure that they have been ethically sourced, do not contain personal or sensitive information, and are appropriate for use in this research study.

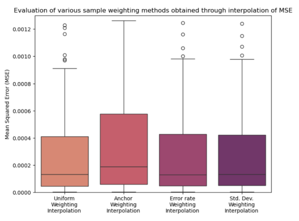
Another vital consideration for this study was the reproducibility of the results. We have published our experiment to GitHub [21]. Finally, we recognize that it is highly important to be aware of the implications this work can have on its research field. Improving machine learning model training efficiency has the potential to reduce computational costs and improve the efficiency of the processes they support. However, we also acknowledge that this improvement in efficiency might unintentionally contribute to applications of machine learning within unethical domains. By making our research, the tools we used and the results we acquired publicly available, there is no moderation in place to ensure these advancements are used responsibly. However, they are intended to support ethical, community-driven research efforts and are designed to do so effectively.

A Monotonicity of Learners in the LCDB 1.0 Dataset

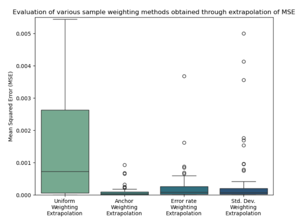
Learner	Flat	Monotone	Convex	Monotone & Convex
SVC_linear	0.00%	95.83%	95.83%	94.44%
SVC_poly	19.44%	80.56%	79.17%	79.17%
SVC_rbf	19.44%	80.56%	76.39%	76.39%
SVC_sigmoid	11.11%	30.56%	56.94%	27.78%
Decision Trees	2.78%	97.22%	97.22%	97.22%
ExtraTrees	2.78%	95.83%	97.22%	95.83%
LogisticRegression	2.78%	97.22%	93.06%	93.06%
PassiveAggressive	1.39%	93.06%	98.61%	93.06%
Perceptron	0.00%	95.83%	98.61%	95.83%
RidgeClassifier	5.56%	76.39%	73.61%	73.61%
SGDClassifier	0.00%	97.22%	100.00%	97.22%
MLP	2.78%	79.17%	70.83%	68.06%
LDA	2.78%	54.17%	50.00%	48.61%
QDA	2.78%	54.17%	58.33%	45.83%
BernoulliNB	19.44%	70.83%	69.44%	65.28%
MultinomialNB	2.78%	95.83%	95.83%	94.44%
ComplementNB	2.78%	90.28%	94.44%	88.89%
GaussianNB	2.78%	79.17%	83.33%	76.39%
KNN	13.89%	84.72%	83.33%	83.33%
NearestCentroid	4.17%	91.67%	93.06%	90.28%
ens.ExtraTrees	8.33%	90.28%	91.67%	90.28%
ens.RandomForest	6.94%	91.67%	93.06%	91.67%
ens.GradientBoosting	0.00%	100.00%	100.00%	100.00%
DummyClassifier	73.61%	23.61%	20.83%	20.83%

Figure 5: Monotonicity of learners in the LCDB 1.0 dataset

B Boxplot Results

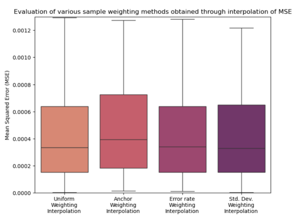


(a) Boxplot comparing interpolation results.

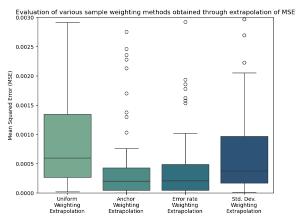


(b) Boxplot comparing extrapolation results.

Figure 6: Comparison of MSE values for Decision Trees using different sample weighting methods (SWMs) for interpolation and extrapolation.

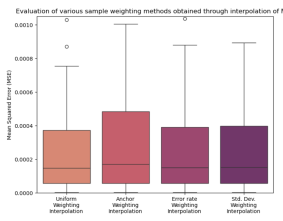


(a) Boxplot comparing interpolation results.

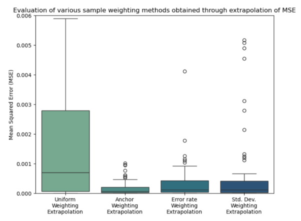


(b) Boxplot comparing extrapolation results.

Figure 7: Comparison of SGD Classifier MSE values for uniform weighting and various sample weighting methods for interpolation and extrapolation.

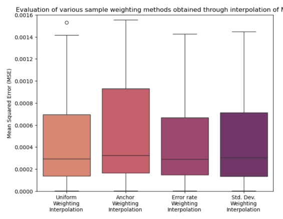


(a) Boxplot comparing interpolation results.

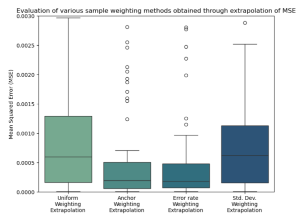


(b) Boxplot comparing extrapolation results.

Figure 8: Comparison of Extra Trees MSE values for uniform weighting and various sample weighting methods for interpolation and extrapolation.



(a) Boxplot comparing interpolation results.



(b) Boxplot comparing extrapolation results.

Figure 9: Comparison of perceptron MSE values for uniform weighting and various sample weighting methods for interpolation and extrapolation.

References

[1] Derek Hoiem, Tanmay Gupta, Zhizhong Li, and Michal Shlapentokh-Rothman. Learning curves for analysis of deep networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4287–4296. PMLR, 18–24 Jul 2021.

[2] Claudia Perlich, Foster Provost, and Jeffrey S. Simonoff. Tree induction vs. logistic regression: a learning-curve analysis. *J. Mach. Learn. Res.*, 4(null):211–255, December 2003.

[3] Tihomir Asparouhov. Sampling weights in latent variable modeling. *Structural Equation Modeling: A Multidisciplinary Journal*, 12:411 – 434, 2005.

[4] ”Tom Viering and Marco Loog”. The shape of learning curves: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7799–7819, 2023.

[5] Felix Mohr, Tom J. Viering, Marco Loog, and Jan N. van Rijn. Lcdb 1.0: An extensive learning curves database for classification tasks. In Massih-Reza Amini, Stéphane Canu, Asja Fischer, Tias Guns, Petra Kralj Novak, and Grigorios Tsoumakas, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2022, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 3–19. Springer, 2023.

[6] Boštjan BRUMEN, Ivan Rozman, Marjan Hericko, Aleš ČERNEZEL, and Marko Hölbl. Best-fit learning curve model for the c4.5 algorithm. *INFORMATICA,*, 25:385–399, 10 2014.

[7] Rosa L. Figueroa, Qing Zeng-Treitler, Sasikiran Kandula, and Long H. Ngo. Predicting sample size required for classification performance. *BMC Medical Informatics and Decision Making*, 12:8 – 8, 2012.

[8] Fulgencio Marín-Martínez and Julio Sanchez-Meca. Weighting by inverse variance or by sample size in random-effects meta-analysis. *Educational and Psychological Measurement - EDUC PSYCHOL MEAS*, 70:56–73, 02 2010.

[9] Jeff Johnson. Factors affecting relative weights: The influence of sampling and measurement error. *Organizational Research Methods*, 7:283–299, 07 2004.

[10] Larry V. Hedges and Jack L. Vevea. Fixed- and random-effects models in meta-analysis. *Psychological methods*, 3(4):486–504, December 1998.

[11] Felix Mohr and Jan N. van Rijn. Learning curves for decision making in supervised machine learning - A survey. *CoRR*, abs/2201.12150, 2022.

[12] Muhammad Abdul Wahab. Interpolation and extrapolation. *Proc. Topics Syst. Eng. Winter Term*, 17:1–6, 2017.

[13] Sayan Mukherjee, Pablo Tamayo, Simon Rogers, Ryan M. Rifkin, Anna Engle, Colin Campbell, Todd R. Golub, and Jill P. Mesirov. Estimating dataset size requirements for classifying dna microarray data. *Journal of computational biology : a journal of computational molecular cell biology*, 10 2:119–42, 2003.

[14] Jyh-Shing Roger Jang and Eiji Mizutani. Levenberg-marquardt method for anfis learning. *Proceedings of North American Fuzzy Information Processing*, pages 87–91, 1996.

[15] Guy Tennenholtz, Tom Zahavy, and Shie Mannor. Train on validation: Squeezing the data lemon, 2018.

[16] M. Neumann, Cecil Hahn, Thomas Horbach, I Schneider, Alexander Meining, Walter Heldwein, Thomas Rösch, and W. Hohenberger. Score card endoscopy: a multicenter study to evaluate learning curves in 1-week courses using the erlangen endo-trainer. *Endoscopy*, 35 6:515–20, 2003.

[17] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[18] Boštjan Brumen, Aleš Černezel, and Leon Bošnjak. Overview of machine learning process modelling. *Entropy*, 23(9), 2021.

[19] Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. Prediction of learning curves in machine translation. In *Annual Meeting of the Association for Computational Linguistics*, 2012.

[20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[21] Lapo den Hollander. research-project-24-25, 01 2025.