# Learning Curve Extrapolation using Machine Learning

## Benefits and Limitations of using LCPFN for Learning Curve Extrapolation

**Pratham Johari** [1]

**Supervisor(s): Mr. Taylan Turan**[1], **Dr. Tom Viering**[1]

[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 28, 2024

Name of the student: Pratham Johari
Final project course: CSE3000 Research Project
Thesis committee: Mr. Taylan Turan, Dr. Tom Viering, Dr. Hayley Hung

## Abstract

This study explores the extrapolation of learning curves, a crucial aspect in evaluating learner performance with varying dataset sample sizes. We use the Learning Curve Prior Fitted Network (LC-PFN), a transformer pre-trained on synthetic data with proficiency in approximate Bayesian inference, to investigate its predictive accuracy using the Learning Curve Database (LCDB). The assessment involves MSE as an error metric, with 2 baselines from previous studies where we see it outperform the baseline in some cases and keep on par in others. Additionally, we scrutinize instances where the LC-PFN model may exhibit shortcomings to identify trends in curve extrapolation failures, offering insights for potential modifications to the training dataset. We see a pattern in learners where LC-PFN performs consistently poorly on, whereas no significant pattern can be seen for datasets.


## 1 Introduction

In the realm of machine learning, one of the factors for the effectiveness of models is the volume of data they are exposed to [1]. However, the acquisition and annotation of data come with considerable cost and time investments. Therefore, determining the optimal amount of data required for a model can be beneficial. Learning curves are a valuable tool in this context, portraying a model's performance against varying sizes of training data. These curves offer insights into how the amount of training data affects models.

Extrapolating the learning curve to predict performance on larger or unseen datasets is a complex challenge that necessitates making assumptions about the curve's shape and behaviour. Existing parametric models, including logarithmic, power-law, and exponential functions, have been proposed for learning curve fitting and extrapolation [2]. Despite their utility, these models may fall short of capturing the intricate and diverse nature of real-world learning curves, potentially leading to inaccurate or unrealistic extrapolations.

To address this extrapolation challenge, our research focuses on leveraging a novel machine learning model: Prior-Data Fitted Networks (PFNs), capable of approximating Bayesian inference [3]. We specifically employ an extended variant, known as Learning Curve PFN (LC-PFN), detailed by Adriaensen et al. in [4]. LC-PFN specializes in predicting model performance in later epochs of training based on earlier epoch performance whereas the learning curves we talk about in this paper compare model accuracy to dataset size. To try to solve our problem, we utilize a dataset of learning curves, referred to as the Learning Curve Database (LCDB) [5]. Using LCDB to test LC-PFN, we aim to analyze if the model can accurately extrapolate learning curves, overcoming the limitations of existing parametric models. An example of extrapolation at different cutoffs can be seen in Figure 1.

In the pursuit of advancing the understanding of learning curve extrapolation, this thesis aims to answer the following research question: *What are the benefits and limitations of using LC-PFN for learning curve extrapolation and how does it compare to other methods?* Through an exploration of this question, we contribute to the evolving discourse on optimizing ML model performance in the face of data constraints.

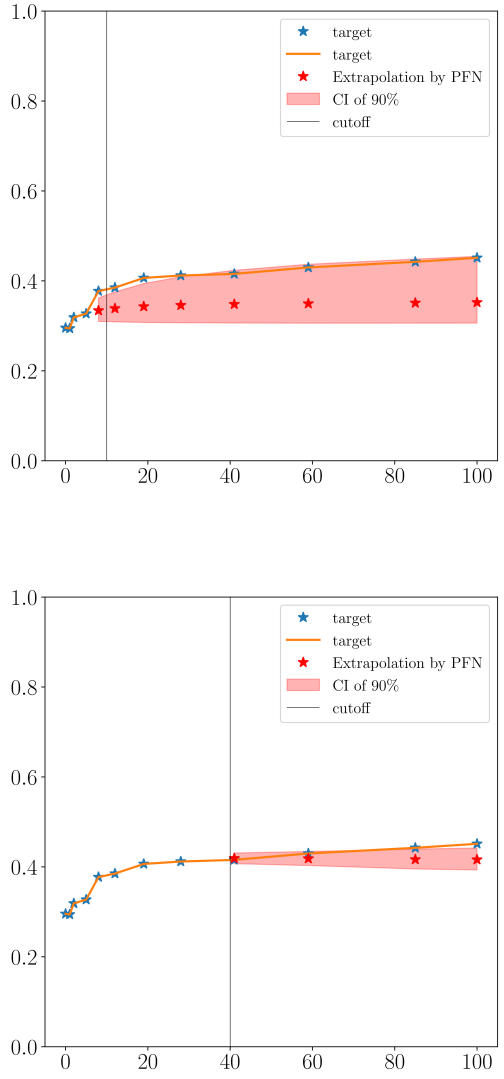


Figure 1: Visualization of curve extrapolation at 10%(*top*) and 40%(*bottom*) cutoff. The Dataset id is 188 and the learner is $SVC\_rbf$

## 2 Related Work

Many papers focus on curve fitting, as evidenced by studies such as [2, 5, 6, 7]. The rationale behind this emphasis lies in the fact that a curve-fitting method can inherently serve as a foundation for effective extrapolation. However, a caveat of curve fitting is its tendency to overlook real-life scenarios where a learner may not conform to the specified parametric model.

Despite this limitation, exploring the performance of various parametric models is valuable for establishing a baseline comparison. In an in-depth analysis by Mohr et al., they scrutinized 16 distinct parametric learning curve models using the same dataset utilized in our work [5]. They identified that $mmf4$ and $wbl4$ outperformed the rest of the parametric models when more than 20% of data was used for fitting. However, at lower percentages, no clear models emerged as superior. Additionally, they found that the $last1$ model performance improves with more data and significantly outperforms the rest at 80% training set. They also saw that parametric models with higher parameter counts performed better overall than those with less.

Kalandadze, in paper [6], concurred with the effectiveness of the $mmf4$ model in fitting learning curves, finding that $mmf4$ them $exp4$ outperformed $pow4$. Conversely, findings by Nguyen in [7] suggest that for classification learners, the $pow4$ models worked best at lower training sets. Nguyen stated that $last1$ takes over as the best-performing model when 80% of the curve is used for fitting like what Mohr et al. found [5]. However, it was emphasized that "*no universal model is confirmed for all tasks, learners, and datasets.*"

Considering these varied conclusions, we limit ourselves to the $last1$ and $mmf4$ models as baselines for comparison against the LC-PFN. The main reason for this limitation is time constraints. By comparing the LC-PFN against these chosen baseline models, we aim to gauge the effectiveness of machine learning in handling the complexities of learning curves, while acknowledging the diversity of tasks, learners, and datasets.

## 3 Methodology

### 3.1 Database

We will utilize the Learning Curve Database (LCDB)[5]. The LCDB comprises 246 datasets from *OpenML* and involves 20 learners from *scikit-learn*. For each dataset size training and testing are repeated 25 times. This implies that for each dataset and learner pair, we have 25 curves representing validation set accuracy, totalling 75 curves (train, validation, and test). However, our focus is solely on the validation set. To preprocess the database, we aggregate curves per dataset and learner pair, following the approach employed by Mohr et al. in [5] in Section 3.3. Additionally, we convert the dataset size for each curve into a percentage where we indicate the max dataset size as 100%, ensuring a consistent framework for the experiments. Please note that the number of points is not the same in each curve. That depends on the true dataset size and differs from dataset to dataset.

### 3.2 Machine Learning Model: LC-PFN

We will employ the Learning Curve Prior Fitted Network (LC-PFN)[4]. A Prior Fitted Network (PFN) is a transformer pre-trained on synthetic data known as a prior. It performs approximate Bayesian inference in a single forward pass. LC-PFN, specifically, is trained on right-censored curves generated from a parametric prior [4]. It is important to note that the curves used to train LC-PFN are different from the learning curves we are researching. In LC-PFN, the curve represents model loss versus epoch, while in our case, we are examining model accuracy versus dataset size. Despite this difference, both curves share similar characteristics and can be fitted with the power law ($pow3$), as Adriaensen et al. utilized as a parametric model to generate the prior [4]. Additionally, Nguyen in [7] has demonstrated that the power law is a suitable model for curve fitting.

Let us look at the model in more detail. The model is supplied with input data—X, Y, and X_test, where X denotes the dataset size up till the cutoff percentage, Y represents accuracy up till the cutoff percentage, and X_test signifies the dataset size for which the model predicts accuracy, i.e., dataset size from cutoff percentage to 100%. We use the functionality provided in the GitHub[1] package to generate the 90% confidence interval (CI). This is done by providing a list of probabilities for which the quantiles are to be predicted. For example, $[0.05, 0.5, 0.95]$ where 5th, 50th, and 95th percentiles.

### 3.3 Model Evaluation Parameters

In evaluating the models, we employ Mean Squared Error (MSE) as our chosen metric, a consistent choice seen in many papers [4, 5, 6]. This decision provides a standardized approach for comparing the performance of different models.

In determining the cutoff percentage, representing the portion of the curve visible to the model for extrapolation, we adopt values of 10%, 20%, 40%, and 80%. This choice adheres to a standardized approach, aligning with percentages commonly employed in numerous studies for curve fitting and extrapolation [4, 5]. Although Mohr et al. in [5] explored additional percentages, we adhere to the specified values due to time constraints. In our experiment, to obtain the points to feed the models, we find the closest point to the specified cutoff percentage and use it as our cutoff point. This method implies that, at times, the cutoff point may be slightly more or less than the labelled percentage. This approach strikes a balance between precision and feasibility within the constraints of our experimentation.

In our analysis, we have incorporated two baseline models: $last1$ and $mmf4$. The former employs a straightforward approach, where it takes the last point

[1] Link: https://github.com/automl/lcpfn

on a given learning curve and always predicts this accuracy value. In contrast, the latter is more intricate, requiring parameter estimation. It used the formula $(ab+cx^d)/(b+x^d)$ where x is the size of the training set and a,b,c,d are parameters to be estimated. The fitting procedure for $mmf4$ is implemented using the code provided by Mohr et al. in [5]. It is important to note that some curve fitting attempts with $mmf4$ result in MSE values greater than 1 and, therefore, are excluded from the curve fitting results. In such cases, these curves are not considered in our analysis to maintain consistency.

### 3.4 Model breakdown

Our primary focus is on identifying scenarios where the LC-PFN model exhibits shortcomings and discussing strategies to improve the representation of real-world conditions in the synthetic training set. We achieve this by scrutinizing whether the target accuracies, i.e., the remaining curve, fall within the model's 90% CI. Specifically, we verify that at least 60% of the target points reside within the 90% CI area. In the graphical representation of extrapolation (Figure 1), both instances show the model breakdown, indicated by the target points outside the 90% CI.

To further understand the nature of the model's shortcomings, we investigate the cases of breakdown by examining each curve individually. We first group the curves by learner and then by dataset, additionally, we check whether the model tends to overestimate or underestimate accuracies for points outside the expected CI. This analysis is carried out by evaluating each target point to determine whether it was underestimated or overestimated by the model. We then label the entire curve based on most of these individual assessments. For example, if most points are found to be above the 90% CI, we conclude that the model has tended to underestimate accuracy for that curve. Again, in graphical representation, in Figure 1 at both cutoffs, the curve is underestimated.

## 4 Results and Discussion

### 4.1 LC-PFN Model Performance Assessment

The performance of the LC-PFN model is evaluated in this section. Table 1 provides an overview of the average MSE values, highlighting the LC-PFN model's competitive performance relative to the baseline models.

Table 1: Average MSE of each model at different cutoff values. Please note that if the curve fitting failed for any baseline, the curve was not included in the overall result.

| Cutoff | 10% | 20% | 40% | 80% |
|---|---|---|---|---|
| $last1$ | 0.0053 | 0.0032 | 0.0016 | **0.0007** |
| $mmf4$ | 0.0091 | 0.0048 | 0.0029 | 0.0022 |
| LC-PFN | **0.0032** | **0.0021** | **0.0013** | 0.0009 |

Remarkably, at a lower cutoff percentage (10%), the LC-PFN model demonstrates superior performance with an MSE of 0.0032, outperforming both $last1$ and $mmf4$. However, it is essential to note that the table alone does not provide the complete picture. Examining the scatter plot in Figure 3, we observe that for $last1$, most points lie below the $x = y$ line. In contrast, this is not the case for $mmf4$, which has more points above the $x = y$ line but also exhibits much higher MSE for several curves. This indicates that although $mmf4$ performs slightly better, it fails to capture the diversity of curves, resulting in a higher overall MSE.

A similar trend is evident in the box plot in Figure 2, where the median for $mmf4$ is lower than LC-PFN, but the mean and overall variance are higher. Despite having a lower average MSE, $last1$ underperforms in variance compared to both $mmf4$ and LC-PFN, as indicated by the outlier; representing the curve with MSE greater than 0.01. Both trends, where $last1$ exhibits high variance and $mmf4$ has a higher mean, persist up to the 80% cutoff where the $last1$ performs better in both the mean and lower outlier. Comprehensive plots for the remaining cutoff percentages are available in the appendix (see Appendix A).

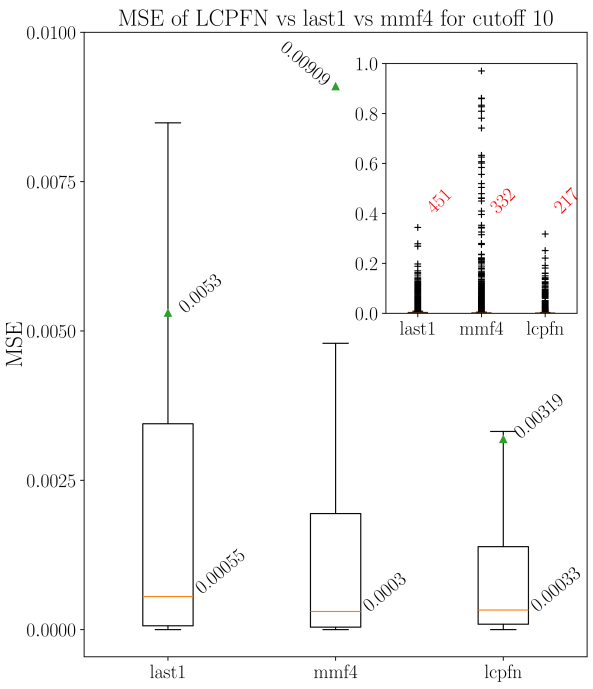


Figure 2: Comparison of MSE of the model and two baseline methods at a 10% cutoff, presented through a box plot. The inner plot displays the full range of the box plot along with outliers, while the outer plot zooms in on values between 0 and 0.01. The red numerical annotation indicates the count of curves with MSE exceeding 0.01, green arrowsheads signify the mean MSE values and the orange line indicates the median value.

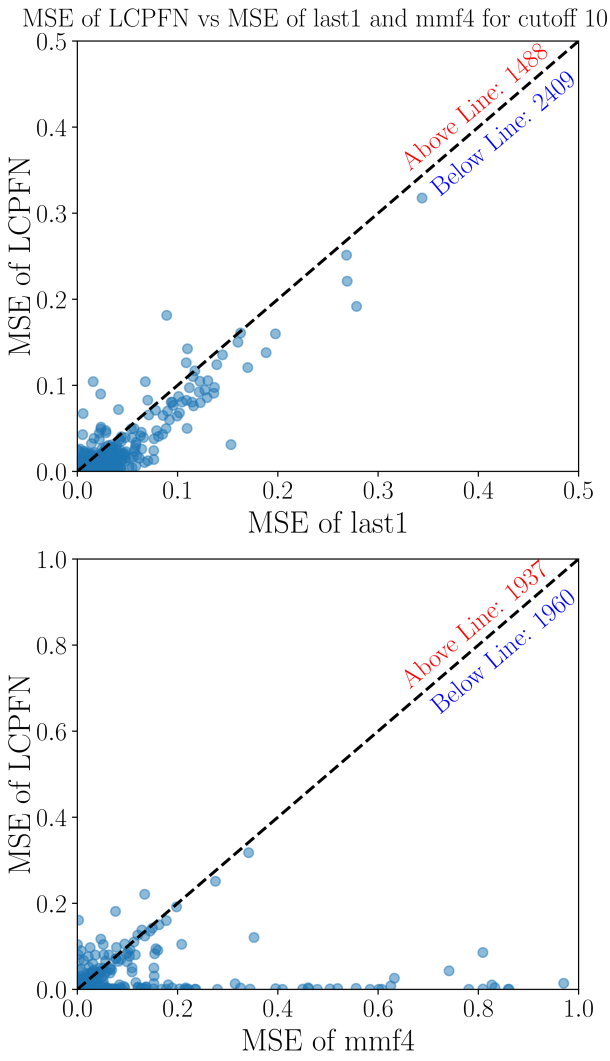


Figure 3: Scatter plot comparing the MSE of extrapolation by the LC-PFN model and the two baselines at 10% cutoff. Each point here represents a curve. The dotted line is $x = y$. *Please note: The scale of the bottom plot is different than the top plot*

### 4.2 LC-PFN Model Shortcomings and Strategies

Our analysis focuses on quantifying the extent to which the LC-PFN model encounters challenges in extrapolating curves. Table 2 provides an overview of the total number of curves that break down the model per cutoff. We notice that the count increases as the cutoff percentage rises. This trend is attributed to the model's heightened confidence in its CI at higher cutoffs, resulting in overconfidence in its accuracy predictions. We believe this is primarily because the model has more points to extrapolate from and has fewer points to extrapolate to. A visual representation of this phenomenon is illustrated in Figure 1.

Table 2: Total count and Percentage of curves per cutoff where the model breaks down and is unable to accurately predict. The total number of curves tested where curves were 4367

| Cutoff | Count | % Out of total |
|---|---|---|
| 10% | 425 | 9.73% |
| 20% | 572 | 13.10% |
| 40% | 986 | 22.58% |
| 80% | 1255 | 28.74% |

Subsequently, we delve into the nuances of how the model breaks down by examining the tendencies of overestimation and underestimation at different cutoffs. Table 3 illustrates a shifting pattern, indicating that at smaller cutoffs, the model tends to overestimate, while at larger cutoffs, it leans towards underestimation.

Table 3: Count of curves based on overestimating and underestimating tendencies of the model at different cutoffs.

| | 10% | 20% | 40% | 80% |
|---|---|---|---|---|
| Under Estimate | 266 | 272 | 406 | 543 |
| Over Estimate | 158 | 296 | 557 | 703 |
| Neither | 1 | 4 | 23 | 9 |
| Total | 425 | 572 | 986 | 1255 |

Finally, we have grouped the curves by learner in Table 4 and by dataset in Table 5. When examining the grouping by learners, a consistent observation emerges — the LC-PFN performs worst consistently on $SVC\_sigmoid$, while the learner where the model performs best on is consistently $sklearn.naive\_bayes.MultinomialNB$ with an exception at cutoff 20% where 19 curves from the learner cause model breakdown. In Appendix A, Figure 6 provides examples of both these learners. The shape of the latter learner aligns with the synthetic data used by the LC-PFN for training, whereas $SVC_sigmoid$ exhibits more unconventional curves.

Looking at Table 5, we see that there are not that many datasets that repeat except for IDs 346 and 1465. In the bottom left of Appendix A, Figure 6, we can see a graphical example of the curves produced by dataset 1465. Due to time constraints, we couldn't investigate the reasons behind the poorer performance of both datasets compared to the others. However, examining the curves suggests that the learners' performance on the datasets is notably unchanging and poor, which could be a contributing factor, though this would require additional verification.

## 5 Conclusions and Future Work

To conclude our research, we explore the application of a novel machine learning model: Prior-Data Fitted Networks (PFNs), which are capable of approximating Bayesian inference to extrapolate learning curves [3]. Specifically, we utilize an extended variant known as Learning Curve PFN (LC-PFN), as detailed by Steven

Table 4: Top 5 and Bottom 5 learners with the highest count of curves that cause model breakdown per cutoff, along with the corresponding counts.

| | **10%** | | **20%** | |
|---|---|---|---|---|
| Top 5 | SVC_sigmoid | 68 | SVC_sigmoid | 99 |
| | sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 54 | sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 54 |
| | SVC_rbf | 34 | sklearn.linear_model.Perceptron | 37 |
| | SVC_poly | 27 | sklearn.neural_network.MLPClassifier | 36 |
| | sklearn.neural_network.MLPClassifier | 25 | SVC_linear | 33 |
| Bottom 5 | sklearn.ensemble.RandomForestClassifier | 9 | sklearn.tree.DecisionTreeClassifier | 17 |
| | sklearn.linear_model.LogisticRegression | 9 | sklearn.ensemble.RandomForestClassifier | 15 |
| | sklearn.tree.DecisionTreeClassifier | 8 | sklearn.ensemble.GradientBoostingClassifier | 15 |
| | sklearn.tree.ExtraTreeClassifier | 6 | sklearn.linear_model.LogisticRegression | 13 |
| | sklearn.naive_bayes.MultinomialNB | 4 | sklearn.ensemble.ExtraTreesClassifier | 11 |
| | **40%** | | **80%** | |
| Top 5 | SVC_sigmoid | 104 | SVC_sigmoid | 114 |
| | sklearn.linear_model.PassiveAggressiveClassifier | 71 | sklearn.linear_model.Perceptron | 101 |
| | sklearn.neural_network.MLPClassifier | 64 | sklearn.linear_model.PassiveAggressiveClassifier | 98 |
| | sklearn.linear_model.Perceptron | 62 | sklearn.linear_model.SGDClassifier | 71 |
| | sklearn.linear_model.SGDClassifier | 56 | sklearn.tree.ExtraTreeClassifier | 70 |
| Bottom 5 | sklearn.ensemble.RandomForestClassifier | 40 | SVC_poly | 47 |
| | sklearn.naive_bayes.BernoulliNB | 36 | sklearn.ensemble.ExtraTreesClassifier | 45 |
| | sklearn.ensemble.ExtraTreesClassifier | 35 | sklearn.linear_model.RidgeClassifier | 43 |
| | sklearn.tree.DecisionTreeClassifier | 35 | sklearn.discriminant_analysis.LinearDiscriminantAnalysis | 39 |
| | sklearn.naive_bayes.MultinomialNB | 31 | sklearn.naive_bayes.MultinomialNB | 35 |

Table 5: Top 10 datasets with the highest count of curves that cause model breakdown per cutoff, along with the corresponding counts.

| 10 | | 20 | | 40 | | 80 | |
|---|---|---|---|---|---|---|---|
| Dataset id | Count | Dataset id | Count | Dataset id | Count | Dataset id | Count |
| 1457 | 18 | **346** | 13 | 41157 | 19 | **346** | 17 |
| 1515 | 10 | 41157 | 11 | 380 | 19 | 18 | 16 |
| 446 | 10 | **1465** | 11 | **1465** | 17 | 299 | 16 |
| 1083 | 10 | 1083 | 11 | 55 | 15 | 1083 | 15 |
| 188 | 8 | 1084 | 11 | 61 | 15 | 1465 | 15 |
| 1088 | 8 | 40677 | 10 | 1450 | 15 | 1499 | 15 |
| 299 | 7 | 336 | 9 | 346 | 13 | 1457 | 14 |
| 40975 | 6 | 1457 | 9 | 41142 | 12 | 1085 | 14 |
| 41161 | 6 | 718 | 9 | 1086 | 12 | 723 | 13 |
| 1233 | 6 | 1088 | 8 | 392 | 12 | 188 | 12 |

Adriaensen et al. in [4]. We evaluate its performance in comparison to two baselines, $las1$ and $mmf4$, described in Section 3.3. LC-PFN exhibited comparatively better performance at lower cutoff values, i.e., the portion of the curve visible to the model for extrapolation, as shown in Table 1. It maintained competitive performance at higher cutoff values as well.

Additionally, we analyzed cases where the model did not perform well to enhance the synthetic dataset used for its training. We observed that the model became more confident with a higher cutoff, but this overconfidence often led to model breakdown. There was a discernible pattern in the learner where the LC-PFN underperformed, particularly *SVC_sigmoid*, revealing the instability of the learner and a weakness for the LC-PFN. Notably, no clear pattern emerged in curves that were failing when grouped by datasets. However, the LC-PFN did perform worse on datasets ID 346 and 1465 in a few cases. The main cause of why the model performed worst on those specific datasets is still unknown and was not explored due to time limitations but looking at the curve for 1465 (bottom left in Figure 6), we see that the learners perform poorly on it, suggesting that this may be a contributing factor.

## 5.1 Future work

In our pursuit of advancing the model's capabilities, preliminary attempts were made to train it using a modified version of synthetic sampling to use the LCDB. Unfortunately, these initial efforts proved unsuccessful, and the associated code can be found in repository[2].

Notably, certain curves using $mmf4$ and/or $last1$ experienced failures where the curve fitting method produces MSE greater than 1. Future research endeavours should try to understand the root causes of these failures and devise strategies for improvement.

The current scaling methodology encounters challenges when applied to curves with too few points, potentially resulting in extrapolation breakdown, even if only one target point falls outside the confidence interval (CI). A similar limitation is observed when dealing with higher cutoffs, where the number of target points is insufficient. In such cases, even a single point can significantly impact the model's ability to make accurate predictions, as reflected in the pattern shown in Table 2. Addressing this concern and refining the scaling ap-

proach are critical areas for future investigation.

Exploration of the impact of altering the minimum percentage of points needed in the confidence interval provided by the model remains an unexplored area. Using percentages other than 60% could offer valuable insights into the model's performance.

Finally, another area for future work could involve determining the CI for $mmf4$ and $last1$; and potentially extending this analysis to other parametric models, enabling another area of comparison with the LC-PFN model. This comparative analysis would contribute to a more comprehensive understanding of each model's strengths and weaknesses.

## 6 Responsible Research

Ensuring the reproducibility of our research is paramount for promoting scientific rigour and transparency. We are committed in providing comprehensive documentation detailing our preprocessing steps, parameters, and data sources. All code utilized in our experiments is accessible to the public[2], facilitating the replication of our findings. We encourage other researchers to validate and build upon our work, fostering a culture of openness in the scientific community

Furthermore, for complete transparency, we acknowledge the use of ChatGPT and GitHub Copilot as tools that supported our research without generating ideas or contributing to data creation. Their role was to assist in error correction and enhance the paper and code's overall coherence. Additionally, we provide the prompts used during the thesis in Appendix B.

## References


[1] Joseph Prusa, Taghi M. Khoshgoftaar, and Naeem Seliya. "The Effect of Dataset Size on Training Tweet Sentiment Classifiers". In: *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. 2015, pp. 96–102. DOI: 10.1109/ICMLA.2015.22.

[2] Tom Viering and Marco Loog. *The Shape of Learning Curves: a Review*. 2022. arXiv: 2103.10948.

[3] Samuel Müller et al. *Transformers Can Do Bayesian Inference*. 2023. arXiv: 2112.10510.

[4] Steven Adriaensen et al. "Efficient Bayesian Learning Curve Extrapolation using Prior-Data Fitted Networks". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=xgTV6rmH6n.

[5] Felix Mohr et al. "LCDB 1.0: An Extensive Learning Curves Database for Classification Tasks". In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Massih-Reza Amini et al. Cham: Springer Nature Switzerland, 2023, pp. 3–19. ISBN: 978-3-031-26419-1.

[6] Anna Kalandadze. *A Comparative Analysis of Learning Curve Models and their Applicability in Different Scenarios*. 2023. URL: http://resolver.tudelft.nl/uuid:571d7746-edef-4b20-83dd-5415f78c5c57.

[7] Dean Nguyen. *In Search of Best Learning Curve Model*. 2023. URL: http://resolver.tudelft.nl/uuid:7921d6fa-b7a3-4fb9-bfdd-cd768da72059.


[2] https://github.com/pratham2442000/BachelorThesis

# A Experiments Additional Plots and Tables

Table 6: Count of curves LCPFN breaks down grouped by learner per cutoff.

| 10% | | 20% | |
|---|---|---|---|
| SVC_sigmoid | 68 | SVC_sigmoid | 99 |
| sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 54 | sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 54 |
| SVC_rbf | 34 | sklearn.linear_model.Perceptron | 37 |
| SVC_poly | 27 | sklearn.neural_network.MLPClassifier | 36 |
| sklearn.neural_network.MLPClassifier | 25 | SVC_linear | 33 |
| sklearn.linear_model.Perceptron | 24 | SVC_poly | 31 |
| sklearn.linear_model.RidgeClassifier | 22 | sklearn.naive_bayes.BernoulliNB | 31 |
| sklearn.linear_model.SGDClassifier | 20 | SVC_rbf | 30 |
| sklearn.linear_model.PassiveAggressiveClassifier | 20 | sklearn.discriminant_analysis.LinearDiscriminantAnalysis | 26 |
| sklearn.naive_bayes.BernoulliNB | 19 | sklearn.linear_model.PassiveAggressiveClassifier | 26 |
| sklearn.discriminant_analysis.LinearDiscriminantAnalysis | 17 | sklearn.linear_model.RidgeClassifier | 23 |
| SVC_linear | 15 | sklearn.neighbors.KNeighborsClassifier | 20 |
| sklearn.neighbors.KNeighborsClassifier | 15 | sklearn.naive_bayes.MultinomialNB | 19 |
| sklearn.ensemble.ExtraTreesClassifier | 15 | sklearn.linear_model.SGDClassifier | 18 |
| sklearn.ensemble.GradientBoostingClassifier | 14 | sklearn.tree.ExtraTreeClassifier | 18 |
| sklearn.ensemble.RandomForestClassifier | 9 | sklearn.tree.DecisionTreeClassifier | 17 |
| sklearn.linear_model.LogisticRegression | 9 | sklearn.ensemble.RandomForestClassifier | 15 |
| sklearn.tree.DecisionTreeClassifier | 8 | sklearn.ensemble.GradientBoostingClassifier | 15 |
| sklearn.tree.ExtraTreeClassifier | 6 | sklearn.linear_model.LogisticRegression | 13 |
| sklearn.naive_bayes.MultinomialNB | 4 | sklearn.ensemble.ExtraTreesClassifier | 11 |
| 40% | | 80% | |
| SVC_sigmoid | 104 | SVC_sigmoid | 114 |
| sklearn.linear_model.PassiveAggressiveClassifier | 71 | sklearn.linear_model.Perceptron | 101 |
| sklearn.neural_network.MLPClassifier | 64 | sklearn.linear_model.PassiveAggressiveClassifier | 98 |
| sklearn.linear_model.Perceptron | 62 | sklearn.linear_model.SGDClassifier | 71 |
| sklearn.linear_model.SGDClassifier | 56 | sklearn.tree.ExtraTreeClassifier | 70 |
| sklearn.neighbors.KNeighborsClassifier | 55 | sklearn.neighbors.KNeighborsClassifier | 66 |
| sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 49 | sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis | 65 |
| sklearn.linear_model.RidgeClassifier | 47 | SVC_linear | 65 |
| sklearn.discriminant_analysis.LinearDiscriminantAnalysis | 47 | SVC_rbf | 64 |
| sklearn.ensemble.GradientBoostingClassifier | 44 | sklearn.ensemble.GradientBoostingClassifier | 63 |
| sklearn.tree.ExtraTreeClassifier | 43 | sklearn.neural_network.MLPClassifier | 62 |
| SVC_rbf | 43 | sklearn.linear_model.LogisticRegression | 54 |
| SVC_poly | 43 | sklearn.tree.DecisionTreeClassifier | 54 |
| sklearn.linear_model.LogisticRegression | 41 | sklearn.ensemble.RandomForestClassifier | 51 |
| SVC_linear | 40 | sklearn.naive_bayes.BernoulliNB | 48 |
| sklearn.ensemble.RandomForestClassifier | 40 | SVC_poly | 47 |
| sklearn.naive_bayes.BernoulliNB | 36 | sklearn.ensemble.ExtraTreesClassifier | 45 |
| sklearn.ensemble.ExtraTreesClassifier | 35 | sklearn.linear_model.RidgeClassifier | 43 |
| sklearn.tree.DecisionTreeClassifier | 35 | sklearn.discriminant_analysis.LinearDiscriminantAnalysis | 39 |
| sklearn.naive_bayes.MultinomialNB | 31 | sklearn.naive_bayes.MultinomialNB | 35 |

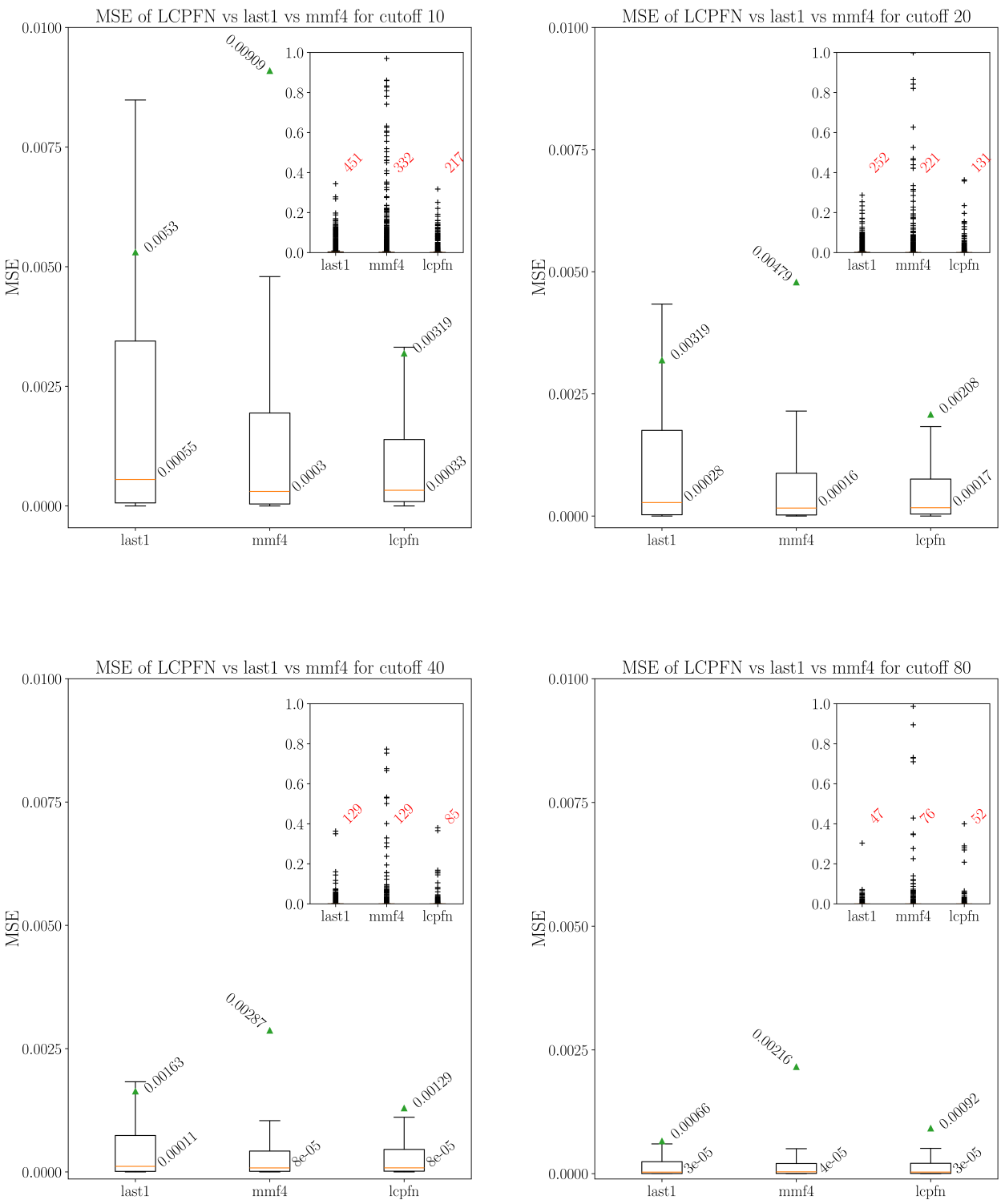


Figure 4: Comparison of MSE among the model and two baseline methods at different cutoff values(top left 10%, 20%, 40%, 80%) presented through a box plot. The inner plot displays the full range of the box plot along with outliers, while the outer plot zooms in on values between 0 and 0.01. The red numerical annotation indicates the count of curves with MSE exceeding 0.01, and a green arrow signifies the mean MSE values.

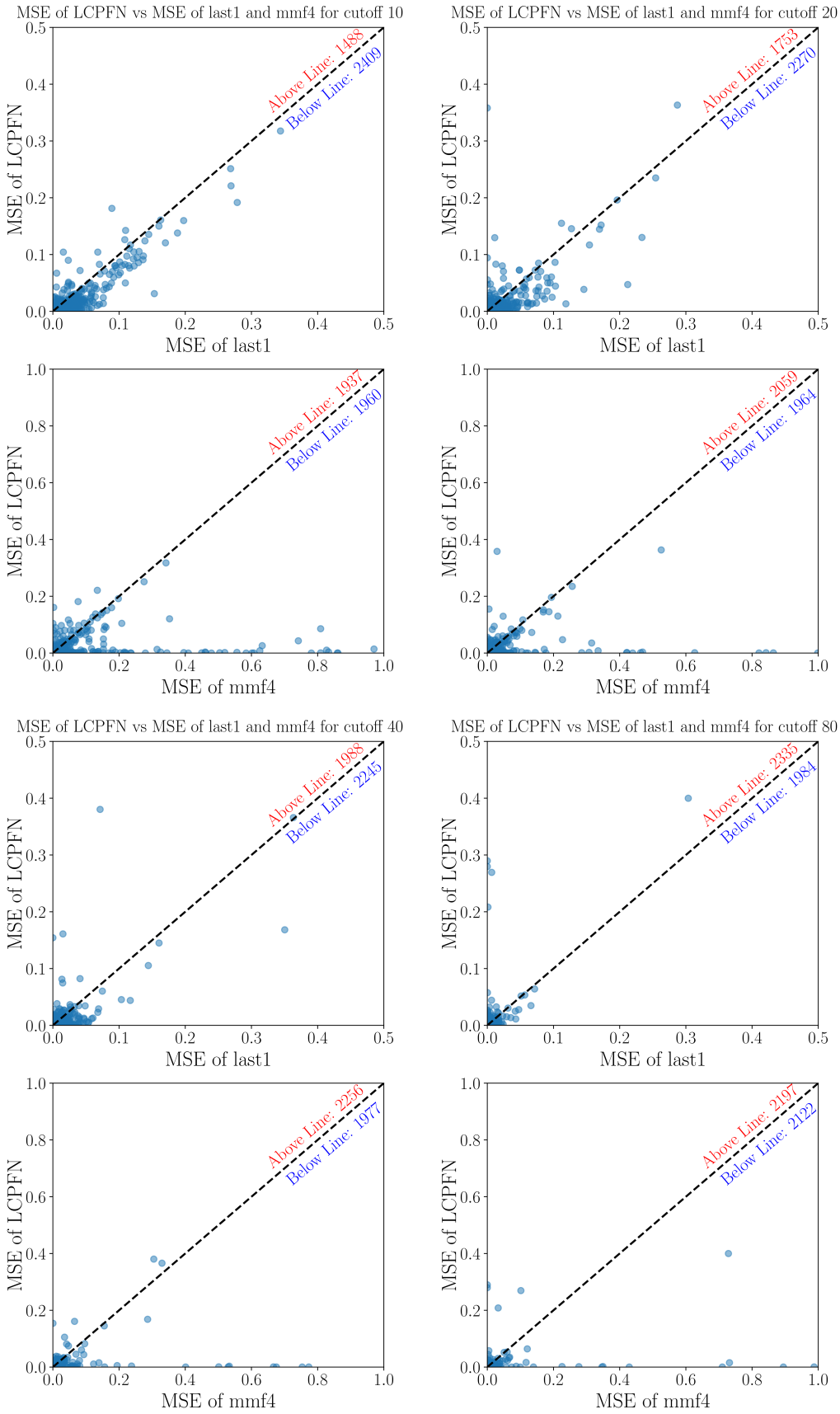


Figure 5: Scatter plot comparing the MSE of extrapolation by the model and the two baselines at different cutoff values. Each point here represents a curve. *Please note: The scale graph with mmf4 is different then of last1*

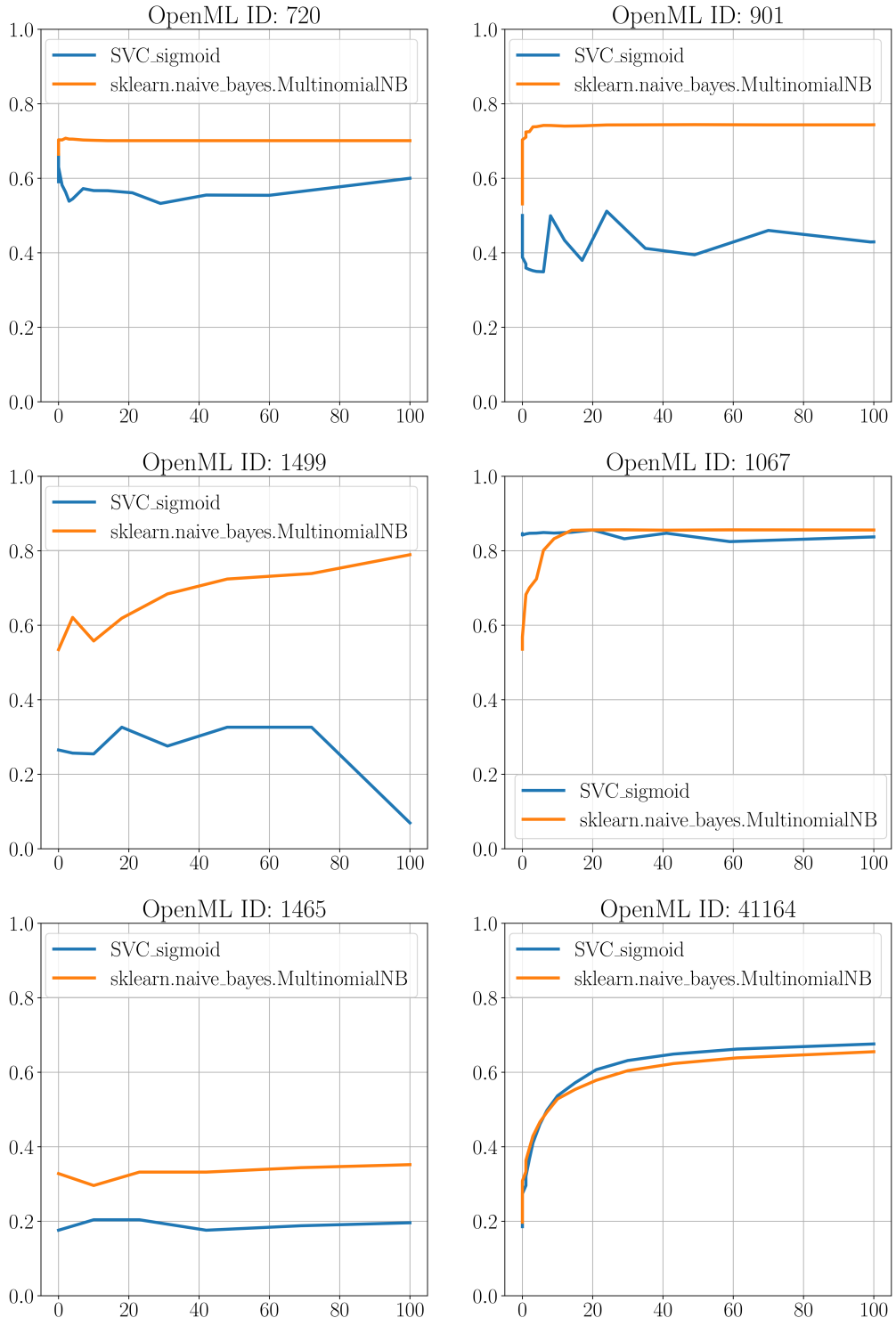


Figure 6: The curves of learner *SVC_sigmoid* and *sklearn.naive$_b$ayes.MultinomialNB* from the same datasets. The LC-PFN model performed the best on the latter in terms of model breakdown, while performing poorly on the former. As we can see, most of the time the curves are quite dissimilar, but there are cases where they are very close, as seen in the bottom right.

Table 7: The Datasets which caused model breakdown per cutoff, grouped by count.

| Cutoff / Count | 10% | 20% | 40% | 80% |
|---|---|---|---|---|
| | OpenML ID | | | |
| 19 | None | None | 380, 41157 | None |
| 18 | 1457 | None | None | None |
| 17 | None | None | 1465 | 346 |
| 16 | None | None | None | 18, 299 |
| 15 | None | None | 55, 61, 1450 | 1083, 1465, 1499 |
| 14 | None | None | None | 1085, 1457 |
| 13 | None | 346 | 346 | 723 |
| 12 | None | None | 392, 1086, 1448, 41142 | 188, 389, 392, 751, 910, 1086, 1448, 1515 |
| 11 | None | 1083, 1084, 1465, 41157 | 54, 299, 393, 395, 740, 797, 1083 | 54, 61, 395, 740, 806, 813, 1087, 1233, 4134, 41159, 42809 |
| 10 | 446, 1083, 1515 | 40677 | 391, 396, 752, 1488, 1499, 4134, 41164 | 31, 55, 401, 718, 799, 912, 1088, 1450, 1488, 40982, 41157, 41164 |
| 9 | None | 336, 718, 1457 | 23, 336, 446, 751, 799, 904, 917, 1087 | 14, 336, 391, 398, 446, 679, 715, 741, 845, 849, 866, 871, 903, 904, 914, 934, 1084, 1479, 1494, 40975 |
| 8 | 188, 1088 | 1088, 1488 | 11, 18, 718, 903, 1041, 1479, 41159 | 11, 21, 23, 181, 743, 913, 1041, 1042, 1468, 41144, 41158 |
| 7 | 299 | 299, 446, 849, 1086, 1087 | 13, 22, 188, 399, 679, 772, 837, 849, 866, 1085, 1475, 1485, 1515, 40664, 40971, 41158, 42809 | 12, 396, 797, 1485, 41142, 41145, 41156 |
| 6 | 751, 1233, 40975, 41159, 41161, 42810 | 13, 715, 1499, 42810 | 181, 398, 741, 743, 806, 871, 979, 1566, 40677, 40981, 41027, 41143, 41144 | 13, 44, 273, 390, 930, 1166, 1566, 40677, 40687, 40981, 40984, 40994, 41150 |
| 5 | 54, 389, 391, 398, 401, 743, 813, 903, 910, 1114, 1448 | 54, 181, 389, 806, 813, 897, 904, 1050, 1112, 1448, 1450, 1468, 1494, 1566, 40971, 41159, 42809 | 14, 273, 389, 390, 845, 930, 1050, 1053, 1084, 1130, 1233, 1441, 1457, 1468, 1494, 41972 | 16, 22, 293, 393, 399, 720, 734, 737, 807, 816, 897, 917, 995, 1120, 1128, 1134, 1161, 1475, 1567, 40498, 40971 |
| 4 | 12, 18, 390, 392, 803, 881, 904, 1465, 1468, 1499, 1592 | 22, 23, 61, 392, 398, 740, 799, 837, 843, 912, 930, 934, 958, 980, 1042, 1114, 1441, 41143 | 12, 16, 44, 60, 401, 722, 807, 913, 991, 1002, 1042, 1067, 1112, 1114, 1590, 1592, 40498, 40687, 40982, 40984, 41145, 41165, 42810 | 6, 26, 36, 201, 354, 722, 735, 752, 761, 772, 837, 846, 847, 962, 1050, 1067, 1112, 1114, 1441, 1464, 1477, 40664, 40978, 41143, 42734, 42810 |

Continued on next page

Table 7: The Datasets which caused model breakdown per cutoff, grouped by count. (Continued)

| | | | | |
|---|---|---|---|---|
| 3 | 13, 21, 201, 395, 734, 740, 866, 912, 913, 917, 958, 1050, 1488, 1503, 1566, 1567, 40978, 41146, 41164, 41972, 42733 | 12, 18, 21, 31, 201, 390, 393, 396, 399, 401, 797, 821, 845, 866, 910, 1000, 1002, 1041, 1085, 1130, 1235, 1464, 40664, 40981, 40994, 41142, 41158, 41972, 42733 | 26, 28, 184, 185, 720, 734, 735, 912, 958, 1000, 1088, 1116, 1128, 1134, 1166, 1235, 1464, 1489, 4538, 40975, 41168, 42733 | 32, 46, 60, 182, 184, 185, 300, 380, 803, 819, 881, 953, 966, 1002, 1116, 1139, 1461, 1483, 1509, 1592, 4538, 40670, 40685, 40701, 40996, 41146, 41163, 41168, 41169, 42733 |
| 2 | 14, 22, 23, 36, 181, 273, 300, 396, 718, 723, 772, 821, 837, 843, 845, 846, 849, 897, 978, 1000, 1002, 1018, 1067, 1112, 1146, 1161, 1235, 1477, 1479, 1509, 40664, 40670, 40981, 40982, 40984, 41144, 41156, 41163, 41165, 42734 | 11, 16, 185, 273, 391, 395, 722, 723, 734, 741, 743, 751, 772, 803, 823, 881, 913, 917, 966, 995, 1019, 1067, 1134, 1139, 1161, 1479, 1483, 1503, 1515, 1590, 1592, 4134, 40687, 40701, 40975, 40982, 40984, 41144, 41146, 41150, 41161, 41164, 41165, 42732, 42734 | 21, 32, 182, 201, 293, 300, 761, 803, 813, 816, 821, 846, 910, 914, 923, 966, 976, 995, 1120, 1138, 1461, 1483, 1487, 1567, 23517, 40668, 40685, 40994, 41146, 41150, 41156, 41161, 42734 | 554, 727, 728, 822, 901, 923, 976, 978, 991, 1000, 1053, 1068, 1111, 1119, 1130, 1138, 1142, 1235, 1487, 1489, 1590, 4137, 4534, 4541, 23512, 23517, 41027, 41161, 41165, 41166, 42732 |
| 1 | 16, 28, 31, 44, 46, 55, 61, 184, 185, 351, 354, 399, 554, 720, 722, 727, 735, 737, 741, 761, 797, 806, 807, 847, 871, 923, 930, 934, 953, 966, 971, 976, 977, 979, 980, 995, 1036, 1041, 1042, 1049, 1053, 1068, 1116, 1119, 1120, 1128, 1130, 1138, 1139, 1142, 1216, 1441, 1450, 1461, 1475, 1483, 1489, 1494, 1575, 4134, 4137, 40498, 40668, 40685, 40701, 40910, 40971, 40996, 41142, 41145, 41150, 41167, 41168, 42732, 42769, 42809 | 14, 28, 36, 38, 44, 46, 180, 184, 188, 300, 351, 354, 380, 679, 720, 727, 735, 737, 761, 807, 819, 822, 846, 847, 871, 903, 914, 923, 953, 959, 971, 976, 979, 1018, 1036, 1040, 1049, 1053, 1056, 1068, 1116, 1119, 1120, 1128, 1146, 1166, 1216, 1233, 1461, 1477, 1485, 1487, 1489, 1509, 1567, 1575, 4137, 4541, 40668, 40685, 40910, 41027, 41156, 41163, 41166, 41167, 41168, 42742, 42769 | 3, 31, 36, 46, 354, 357, 715, 723, 727, 728, 737, 823, 833, 843, 847, 881, 897, 934, 953, 962, 971, 977, 978, 1019, 1020, 1021, 1036, 1049, 1068, 1069, 1139, 1142, 1161, 1216, 1509, 1575, 4135, 4137, 4541, 23512, 40670, 40701, 40983, 41163, 41166, 41167, 42732, 42742, 42769 | 3, 28, 30, 179, 180, 357, 821, 823, 833, 843, 958, 971, 979, 993, 1020, 1036, 1040, 1049, 1069, 1216, 1575, 1597, 4135, 40672, 41228, 41972, 42769 |

## B LLM usage and prompts

The following are the prompts used during the thesis in no particular order.

1. Analyse this text for grammatical error <Text>
2. What is colspec in this table <latex table code>
3. Suggest a way to make these para flow better into each other <Text>
4. Do a grammar check and list the lines where there are issues <Text>
5. <Code>Explain this line by line
6. Can you give me a synonym for this "Model failure", is more related to machine learning where a model is not able to perform its task and fails to work as expected
7. <Style file>How can I change this to put the reference based on appearance instead
8. Modify this caption such that is suitable for a research paper. It needs to be concise and clear <Text>
9. Optimise this <Code>

10. <Code>
11. I want to add the count of points below and above the x=y line <Plotting code>
12. I am writing a paper, follow this style of writing <Text>and give ideas for this <Text>to be clear and concise.
13. <Python error>.
14. I want to plot a bar graph, I get a list of values, but I don't know the range. Suggest ideas for it.
15. I have these two lists which represent points on a curve how can interpolate it?
16. randomly select x% of the train set without changing the ratio of the classes