Scaling Generative Tabular Learning for Large Language Models

Yiming Sun*
University of Pittsburgh
yis108@pitt.edu

Xumeng Wen
Microsoft Research Asia
xumengwen@microsoft.com

Shun Zheng
Microsoft Research Asia
shun.zheng@microsoft.com

Xiaowei Jia
University of Pittsburgh
xiaowei@pitt.edu

Jiang Bian
Microsoft Research Asia
jiang.bian@microsoft.com

Abstract

Developing predictive models for tabular data is essential across many industrial applications. The primary challenge in addressing these tasks lies in handling heterogeneous data schemas and diverse prediction targets. Recently, generative tabular learning (GTL) was developed to leverage the instruction-following paradigm of large language models (LLMs) to enable universal tabular learning across varied datasets. This method facilitates effective prompt-based transfers to downstream tasks without the need for supervised tuning. However, the full potential of GTL-enhanced LLMs remains largely unexplored due to limitations in dataset size, sequence length, and model architecture, leading to notable performance gaps compared to traditional tuning-based tabular models as the number of training examples increases. In this study, we aim to unlock the full potential of GTL from a scaling perspective. We expanded the pre-training datasets from 340 to 972, extended the sequence length from 4,096 to 16,384 tokens, and experimented with different base LLMs. Our findings reveal that scaling datasets and prediction tasks generally enhances generalization, although regression tasks tend to reach saturation quickly. Increasing the number of in-context samples consistently improves performance, especially during inference. Our optimized LLMs demonstrate significant improvements, effectively closing the gap with and even surpassing highly-optimized models when dealing with larger training samples.

1 Introduction

Developing predictive models for tabular data is a fundamental demand across various industrial applications, such as clinical diagnostics and prognostics [1], property price predictions [2], and product sales forecasting [3]. The challenge lies in coping with heterogeneous data schemas across different tabular datasets, which often feature varying columns and data types, as well as handling diverse prediction targets, such as distinct classification categories and regression objectives. Consequently, most existing methods tend to develop a specialized supervised model that is tightly bound to each individual tabular dataset [4, 5, 6, 7, 8, 9, 10]. This approach can become a significant burden when dealing with massive datasets and numerous tasks.

To move towards cross-table tabular learning, recent studies have begun exploring new architectures and pre-training techniques across heterogeneous datasets [11, 12, 13, 14, 15, 16]. Nonetheless, these

*This work was done during the internship at Microsoft Research Asia.

Table Representation Learning Workshop at NeurIPS 2024

approaches still rely on supervised fine-tuning to adapt to specific downstream datasets, typically involving non-trivial hyper-parameter tuning and optimization for each case to secure performance.

Recently, a novel approach called generative tabular learning (GTL) has been introduced [17], aiming to integrate the popular instruction-following paradigm of modern large language models (LLMs) [18, 19, 20] into tabular data learning. GTL advocates for the continual pre-training of LLMs on extensive, language-formatted tabular data using a next-token prediction objective [18]. This enables universal tabular learning across heterogeneous datasets and, more significantly, allows for simple yet effective prompt-based transferring to downstream tasks, breaking the necessity of supervised tuning. According to [17], LLMs enhanced with GTL have shown impressive capabilities in directly handling heterogeneous datasets, performing both zero-shot and in-context (few-shot) learning effectively, and supporting a variety of classification and regression tasks.

However, the potential of GTL-enhanced LLMs remains largely unexplored, as the models developed by [17] were only pre-trained on 340 Kaggle datasets, limited to a sequence length of $4,096$ tokens, and confined to the LLaMA [21] architecture. This limitation restricts the effectiveness of their prompt-based transfer learning in broader downstream applications, particularly those that involve more training examples and call for longer sequence lengths. For instance, while these LLMs demonstrate state-of-the-art performance in extremely few-shot scenarios, they significantly underperform typical tree-ensemble models as the number of shots reaches 64.

In this work, we address this limitation by further scaling the GTL process, aiming to replicate the success obtained in scaling LLMs [22, 23] and fully unlock the potential of GTL. Unlike vanilla LLM pre-training, where data scale can simply be represented by the number of tokens, GTL involves formatting instruction-oriented tabular data as a sequence of language tokens and imposes loss calculations only for numerical feature and label tokens. Thus, we regard a single tabular data example that involves loss calculations as a basic element when considering data scaling.

Moreover, since tabular data encompasses different datasets, intra-dataset samples share the same data schema and follow the same distribution, while inter-dataset samples likely have very different features and cover distinct patterns. Therefore, it is essential to distinguish between the number of unique datasets in total and the number of samples (shots) within each dataset. Furthermore, even within the same dataset, assigning different columns as prediction targets results in different tasks. Samples from different tasks within the same dataset are more closely related than samples from different datasets, necessitating a differentiation between the number of tasks per dataset. Additionally, another factor to consider is the number of in-context samples, which can range from zero (zero-shot) to an arbitrary large number, as long as the maximum sequence length permits. Besides these data factors, we also experiment with different base LLMs to optimize our approach.

With these factors in mind, we significantly scale the GTL process described in [17]. Specifically, we increase the number of pre-training datasets from 340 to 972 and extend the maximal sequence length from $4,096$ tokens to $16,384$. Additionally, we experiment with two LLM families, LLaMA-2 [21] and Phi-3 [24]. Through extensive experiments, we have obtained several interesting findings and produced significantly improved GTL-enhanced LLMs.

- Scaling datasets and increasing prediction tasks per dataset generally follow a power-law relationship, contributing to generalization performance on held-out classification tasks. However, generalization capabilities on held-out regression tasks quickly saturate and begin to oscillate with further scaling. We conjecture that this is because regression tasks are more challenging, involving multi-step next-token predictions.
- Unfortunately, scaling the number of samples per dataset and per task hits an apparent ceiling very early for both classification and regression. We speculate this is due to redundancy, as more samples from the same task distribution bring fewer distinctive patterns.
- The good news is that increasing the number of in-context samples allowed during pre-training contributes to steady improvements in in-context learning for held-out tasks, applicable to both classification and regression. Specifically, all GTL-enhanced LLMs benefit significantly from the increase in in-context samples during inference.
- Another interesting observation is that model scale plays a more critical role in regression than in classification. Additionally, we find that the Phi-3 series excel in regression tasks compared to the LLaMA-2 series. We speculate that this may be related to differences in their pre-training data, although this is difficult to confirm.

- Finally, by combining the best configurations of all scaling factors, we have developed new GTL-enhanced LLMs with significantly improved performance. Notably, when the number of training examples exceeds 64, our single-forward inference can still perform on par with or even surpass highly optimized tabular models with only a single forward inference.

2 Related work

Paradigm evolution in tabular data learning We begin by providing a concise overview of the evolution of tabular data learning paradigms. The most prominent approach has concentrated on extracting crucial features or developing effective representations to enhance prediction tasks. This category encompasses early tree-ensemble methods [4, 5, 6] and, with the advent of deep learning [25], later neural models [26, 27, 28, 7]. The debate continues over whether tree-based models or neural models are superior for tabular data [8, 10]. Unlike traditional tree-based models, which are optimized independently for each tabular learning task, modern neural tabular models have evolved with advanced mechanisms. These include the introduction of self-supervised learning objectives [29, 30, 31, 32], cross-table learning [11, 12, 13, 14, 15, 16], and evaluating zero-shot [11, 33] and in-context [34, 35] learning capabilities. More recently, there has been a trend towards integrating language models with tabular data learning [34, 33, 15, 17, 16].

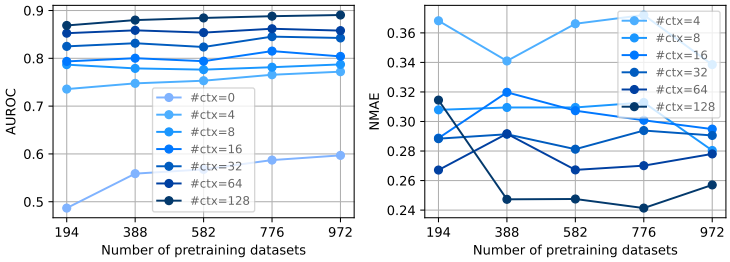
When language models meet tabular data learning Several attempts have been made to combine language models with tabular deep learning due to their unique advantages in encoding categorical feature values, understanding semantic feature meanings, and providing prior knowledge. For instance, LIFT [34] explored the tabular data learning capabilities of proprietary GPT models [19]. TabLLM [33] fine-tuned the T0 model [36] on various tabular classification datasets and observed superior performance in few-shot scenarios. Despite achieving impressive results, these studies directly applied a base language model to a tabular learning task without exploring cross-table learning, thereby missing the opportunity to develop universal tabular learning capabilities. Subsequent methods, such as TP-BERTa [15] and CM2 [16], have investigated cross-table pre-training using customized BERT variants [37], but they still require fine-tuning to adapt to new learning tasks. Distinct from these earlier efforts, GTL [17] has both performed cross-table pre-training and made significant progress towards simple yet effective prompt-based adaptation for new tasks, initially using the LLaMA series [21] for demonstrations. Therefore, we adhere to the GTL paradigm in this work, which provides a more practical and user-friendly user interface in inference. Our focus is to further enhance its performance from a scaling perspective.

Scaling language models Scaling is a critical factor in the development of modern LLMs, contributing to their transformation into versatile generalists [18, 19, 21, 38]. To maximize the use of limited computational resources and train LLMs to their fullest potential, researchers have devised pre-training methodologies known as "scaling laws" [22, 23]. The fundamental insight behind these laws is the power-law relationship between both data and model scales and their generalization performance, allowing for predictive extrapolation. In this work, we aim to uncover analogous scaling laws for GTL. However, this endeavor presents unique challenges, such as accounting for the number of datasets, the number of tasks per dataset, the number of shots per dataset and per task, and the number of in-context options. Additionally, we need to investigate which types of base LLMs are most suitable for tabular data learning.

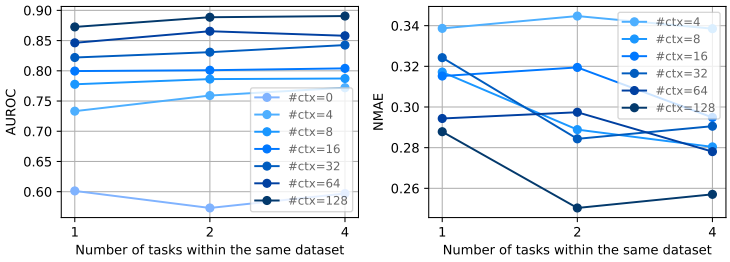
3 Data Construction for GTL

In this work, we introduce a hierarchical scaling approach to integrate tabular data with LLMs. The scaling process operates as follows: We begin with a collection of datasets from various domains to ensure diversity. For each dataset, we select multiple columns as labels to create different tasks. This allows us to explore various prediction objectives within the same dataset. For each task, we determine the number of in-context samples. Depending on whether we're performing zero-shot or in-context learning, we generate different templates which will be elaborated later. Finally, we decide how many shots (training examples) to generate for each setting. This step allows us to scale the amount of pre-training data.

The total size of the pre-training dataset is calculated using the following equation:



(a) Dataset scaling



(b) Task scaling

Figure 1: Aggregated performance on held-out classification and regression tasks by scaling datasets and tasks, where we differentiate different numbers of in-context examples (#ctx) used for inference.

$$N = N_{Dataset} \times N_{Task} \times N_{Context} \times N_{Template} \times N_{Shot},$$

where N, $N_{Dataset}$, N_{Task}, $N_{Context}$, $N_{Template}$, N_{Shot} represent for the total dataset size, number of datasets, number of tasks per dataset, number of context examples, number of templates, number of shots respectively.

This hierarchical scaling strategy enables us to systematically expand the dataset and thoroughly investigate the effects of each component on the model's ability to learn from tabular data, which will be elaborated in Section 4.

4 Scaling experiments and analyses

In this section, we illustrate how we scale our tabular dataset for LLMs at the dataset level (Section 4.1), context level (Section 4.3), sample level (Section 4.2), and model level (Section 4.4). We have conducted a series of experiments for each aspect and analyzed the effect of each scaling method on the performance, evaluated using the Area Under the Receiver Operating Characteristic curve (AUROC) for classification tasks and the Normalized Mean Absolute Error (NMAE) for regression tasks. Regression tasks are more challenging than classification tasks because they require predicting a sequence of digits, which can lead to unstable results. For both types of tasks, we examine zero-shot and in-context learning scenarios. Zero-shot evaluation is extremely difficult because we evaluate the model on holdout datasets entirely different from the pre-training datasets.

4.1 Scaling datasets and tasks

Scaling the number of datasets involved in pre-training is one of the most intuitive approaches when working with tabular data. By increasing the quantity of datasets, the model is exposed to a broader range of data areas, feature types, and distributions, including larger collections of categorical feature values and broader ranges of numerical values. This enriches the model's understanding of diverse tabular data patterns.

To build a diverse and comprehensive dataset for our experiments, we downloaded and processed 156 classification and 184 regression datasets from Kaggle[2], as well as 505 classification and 127 regression datasets from OpenML[3]. This extensive collection spans a wide range of domains and feature types. Additionally, we incorporated 18 classification and 18 regression datasets as holdout sets for evaluation purposes, with detailed information provided in the Appendix A.1. Besides, we follow the official implementation[4] to run pre-training and perform held-out evaluation.

To investigate how the quantity of collected datasets influences performance, we conducted experiments by training models on different numbers of datasets. As illustrated in Figure 1(a), we observed that steadily increasing the number of datasets involved in the pre-training process leads to an overall improvement in performance on both holdout classification and regression datasets. This trend underscores the significance of dataset diversity and volume in enhancing the model's ability to generalize to unseen data. The performance improvement is more pronounced in zero-shot learning and few-shot learning scenarios with limited context. This is likely because, with more pre-training datasets, the model learns more universal and robust underlying patterns inherent in tabular data. These patterns provide a foundational understanding that compensates for the lack of in-context examples, enabling the model to make more accurate predictions even when contextual information is absent or minimal.

Given a limited number of datasets, we aim to generate various types of prompts that effectively utilize the available data. Following the method outlined in [17], the first approach is to generate multiple templates to represent features in tabular data. Similar to natural language, which can be naturally applied to LLMs, the T-lang template is introduced to convert each feature into a sentence, with meta-information provided at the beginning of the prompt. While this approach ensures detailed representation, it often results in repetitive descriptions and long prompts, which can be inefficient for in-context learning. To address this, the T-table template is introduced, which uses a Markdown table format to structure the tabular data, placing feature meanings at the start of the table. This method provides a more concise and effective way of handling tabular data. Furthermore, the T-anony template, a variant of the T-table format, omits all meta-information, simulating practical scenarios where the background and feature meanings of tabular data samples are unknown. This template focuses solely on the data itself, offering a approach for cases with limited prior knowledge. In our experiments, we employ the T-lang and T-table templates for zero-shot learning, leveraging their distinct ways of expressing tabular data features. For in-context learning, we utilize the T-table and T-anony templates, with the former providing structured feature descriptions and the latter simulating scenarios where feature meanings are unknown, focusing solely on the raw data itself.

To further enrich the diversity of pre-training datasets, we generated up to four unique tasks for each dataset by strategically selecting different columns as labels, while using the remaining columns as features. To examine the effect of increasing the number of tasks on model performance, we designed a series of experiments where models were trained on data with varying numbers of tasks: 1 task, 2 tasks, and 4 tasks. As shown in Figure 1(b), even without adding more datasets to the continual pre-training process, we observed a slight increase in performance. Given that the features in our datasets are mostly numerical, expanding our training datasets through task scaling primarily introduces additional regression tasks. Consequently, while the performance gains in classification tasks are modest, the increase in performance is more pronounced and noteworthy in regression tasks, although the results exhibit some instability.

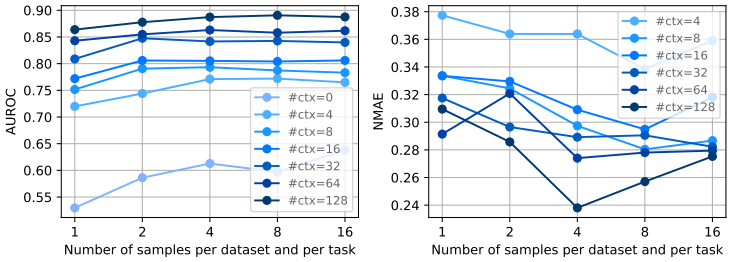
4.2 Scaling examples per dataset and per task

While LLMs can learn data patterns effectively from a number of examples, achieving an optimal balance in the sample size is crucial for maintaining strong performance. In sample level scaling, a small set of examples—commonly referred to as "shots"—are used to help the model grasp the underlying structure and relationships within the data, enhancing the model's ability to generalize across different tasks. However, including too many samples during training can lead to data duplication, where one instance in dataset might be selected several times for training in random selection. This duplication harms the independence of instances, leading to a bias where true diversity and distribution of the dataset are not accurately reflected.

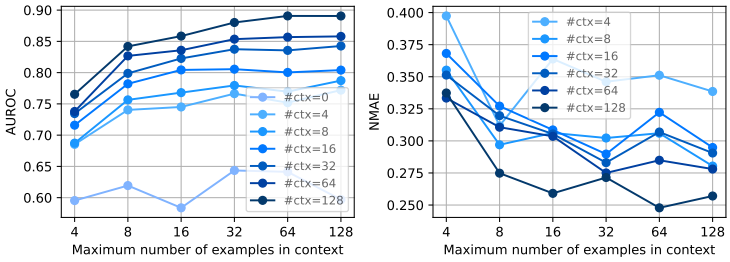
[2] `https://www.kaggle.com/datasets`

[3] `https://www.openml.org/search?type=data`

[4] `https://github.com/microsoft/Industrial-Foundation-Models`



(a) Sample scaling



(b) Context scaling

Figure 2: Aggregated performance on held-out classification and regression tasks by scaling samples per dataset and per task as well as in-context examples in pre-training, where we also differentiate different numbers of in-context examples (#ctx) used for inference.

Sample scaling is the most delicate scaling method in our work. As shown in Figure 2(a), when we select up to 4 samples for each setting, the performance increases as the number of samples grows. However, once the number of samples reaches 4, further increasing the number leads to a plateau in classification performance and even a decline in regression performance. This suggests that, in our dataset, the model saturates when the number of samples reaches 4. Adding more samples beyond this point incurs additional computational cost without improving model performance.

4.3 Scaling in-context samples in pre-training

Scaling up with context is crucial for improving the performance of LLMs when working with tabular data. In-context learning, where models are provided with relevant examples or auxiliary information during inference, enables the model to make more informed decisions by utilizing patterns from similar data points. Tabular data, often characterized by its structured nature and complex interrelationships between features, benefits significantly from additional contextual information, as it allows the model to better understand the dependencies between features. By increasing the amount of context provided to the model, we enhance the model's ability to generalize across varied datasets and adapt to new tasks.

Building on [17], our work scales maximum number of in-context examples to 128, resulting in seven in-context scenarios: {0, 4, 8, 16, 32, 64, 128}. While increasing the number of in-context examples provides the model with richer context, it also introduces the challenge of longer prompt lengths. This can become a limiting factor for some LLMs, especially those with constraints on input length, requiring careful consideration in practical applications.

To address the issue of long prompt lengths, which are capped at $16,384$ tokens in our experiments, we utilize RoPE embeddings [39] instead of absolute or relative position embeddings. RoPE embeddings have demonstrated superior performance when handling long sequences. Additionally, following the approach in [38], we decrease the rotation angle by increasing the hyperparameter "base frequency b" from $10,000$ to $500,000$, which helps mitigate the decaying effect of RoPE on distant tokens, enhancing the model's ability to process longer sequences.

We conducted a series of experiments by increasing the maximum number of context examples in pre-training, as shown in Figure 2(b). We observed that the overall performance on both classification

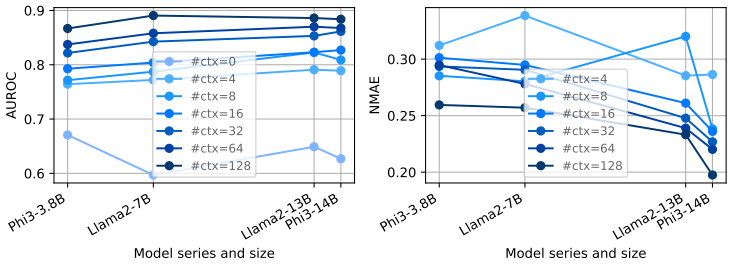


Figure 3: Aggregated performance on held-out classification and regression tasks by using different base LLMs, where we differentiate different numbers of in-context examples (#ctx) used for inference.

and regression tasks improved significantly as we expanded the context length. Specifically, when models were trained only on zero-shot learning and in-context learning with 4 context examples, we found that in-context learning with contexts larger than 4 outperformed the setting with 4 context examples, demonstrating the extrapolation ability of LLMs for tabular data. Additionally, adding longer contexts during training also enhanced performance on shorter contexts. These findings indicate that performance across different context lengths influences one another.

4.4 Scaling base LLMs

Scaling LLMs at the model level, primarily by increasing the number of parameters, has been shown to significantly enhance their ability to handle a wider range of tasks. Larger models are capable of capturing more complex patterns and relationships in data, making them better suited for tasks that require deep understanding and nuanced decision-making. However, this scaling comes with challenges, such as increased computational and memory requirements, as well as longer training times. Despite these challenges, model level scaling remains a critical avenue for expanding the capabilities of LLMs, allowing them to process more sophisticated datasets, including those with complex tabular structures.

In this work, we build our models by continually pre-training from LLaMA2-7B/13B checkpoints, as well as Phi3-3.8B/14B checkpoints (specifically, Phi-3-mini-instruct-128K and Phi-3-medium-instruct-128K, respectively). By comparing these four models together in Figure 3, we observe that larger models are better at understanding tabular data, leading to improved performance in both classification and regression tasks. Notably, the Phi-3-3.8B model already demonstrates strong performance on classification tasks despite its relatively smaller size compared to the other models. However, in regression tasks, increasing the model size significantly enhances performance. Specifically, the NMAE measurement for in-context learning with contexts larger than 8 shows a monotonic decreasing trend as model size increases—a pattern not observed in other scaling methods due to the typically unstable nature of regression tasks. We attribute this result to the inherent differences in task difficulty for LLMs: classification tasks require predicting a single digit representing a category, whereas regression tasks involve multi-step predictions to generate continuous numerical outputs.

5 Comparative analyses

By scaling our tabular dataset at the dataset level, context level, and sample level, and also scaling the base LLM at the model level, we propose a series of LLMs termed Generative Tabular Learning with Scaling (GTL-S). To demonstrate the effectiveness of our GTL-S models, we conduct a relative performance evaluation with GTL in Section 5.1, analyzing the improvements achieved through our scaled tabular datasets. Furthermore, we perform an absolute performance evaluation against competitive tabular models in Section 5.2 to showcase the competence of our GTL-S models.

5.1 Relative performance evaluation with GTL

To gain a deeper understanding of how scaling tabular datasets improves the performance of LLMs, we conducted a detailed comparative analysis on evaluation datasets, examining each dataset per context count. As illustrated in Figure 4, we arranged the evaluation tasks according to the length

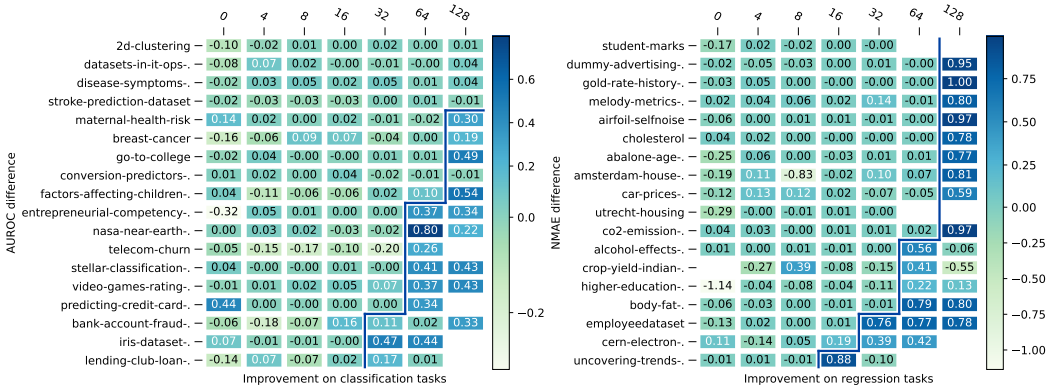


Figure 4: Performance comparisons between our optimized LLaMA2-13B-GTL-S and LLaMA2-13B-GTL released by [17], where we calculate performance improvements in AUROC or NMAE and use a blue line to indicate the boundary of 4K-tokens sequence length.

of sequences for each dataset and the number of context samples included. The values presented represent the differences in evaluation metrics—specifically, the AUROC for classification tasks and the NMAE for regression tasks. In previous GTL models, input sequences were truncated at 4,096 tokens, however, our GTL-S models extend the maximum sequence length to 16,384 tokens. To better illustrate the improvements achieved by our GTL-S models, we included broken lines in the figure; tasks represented below these lines have sequences exceeding 4,096 tokens. As is evident from the figure, there are significant improvements in performance for both classification and regression tasks when using GTL-S models. Importantly, for tasks with sequences shorter than 4,096 tokens, both GTL and GTL-S models perform at an equivalent level. This observation indicates that GTL-S models enhance performance on long-context tasks without sacrificing accuracy on short-context tasks, effectively demonstrating their ability to handle extended sequences while maintaining robustness across different context lengths.

5.2 Absolute performance evaluation against competitive tabular models

To demonstrate the competence of our GTL-S models in tabular learning, we performed an absolute performance evaluation against competitive tabular models (detailed in the Appendix A.2), as shown in Table 1 and Table 2. For zero-shot learning, LLMs fill the vacancy in supervised tabular learning, as traditional models are incapable of zero-shot inference. In few-shot learning scenarios where the number of context examples in evaluation is less than or equal to 16, both GTL and GTL-S models exhibit significantly better performance than supervised tabular models. Unlike the performance decrease observed in GTL [17], our GTL-S models maintain competitive performance even when the number of context examples is greater than or equal to 32. They remain on par with outstanding baselines such as TabPFN, CatBoost, and XGBoost, despite requiring only a single forward inference. By comparing the performance with the original LLaMA2 and Phi3 models, we confirm that the tabular learning abilities of GTL-S models are not merely inherited from their base models. Notably, the original Phi3 models demonstrate better tabular learning abilities than LLaMA2, likely due to differences in their pre-training datasets.

6 Conclusion

In this work, we introduce a series of Generative Tabular Learning with Scaling (GTL-S) models, designed by scaling across several dimensions, including datasets, tasks, examples per dataset and task, in-context samples, and the base LLMs. We notably increased the number of pre-training datasets from 340 to 972 and extended the sequence length from 4,096 to 16,384 tokens. The GTL-S models exhibit marked improvements in long-context tabular predictions, consistently matching or surpassing the performance of traditional, highly optimized tabular models, even when working with large context examples. These findings highlight the strong potential and capability of the GTL-S models in addressing complex tabular data challenges.

Table 1: Averaged AUROC subscripted with 25% and 75% quantiles on held-out classification tasks.

Model	0	4	8	16	32	64	128
LightGBM	–	$0.50_{[0.50,0.50]}$	$0.50_{[0.50,0.50]}$	$0.50_{[0.50,0.50]}$	$0.50_{[0.50,0.50]}$	$0.82_{[0.70,0.97]}$	$0.88_{[0.82,0.98]}$
XGBoost	–	$0.54_{[0.50,0.51]}$	$0.67_{[0.50,0.79]}$	$0.76_{[0.66,0.92]}$	$0.80_{[0.68,0.95]}$	$0.86_{[0.77,0.96]}$	$0.88_{[0.85,0.98]}$
CatBoost	–	$0.73_{[0.61,0.89]}$	$0.78_{[0.69,0.90]}$	$0.81_{[0.73,0.93]}$	$\mathbf{0.85}_{[\mathbf{0.74},\mathbf{0.97}]}$	$\mathbf{0.87}_{[\mathbf{0.82},\mathbf{0.98}]}$	$0.89_{[0.87,0.98]}$
FTTransformer	–	$0.63_{[0.53,0.76]}$	$0.65_{[0.54,0.76]}$	$0.70_{[0.59,0.83]}$	$0.67_{[0.53,0.81]}$	$0.66_{[0.50,0.82]}$	$0.68_{[0.53,0.80]}$
XTab	–	$0.55_{[0.43,0.69]}$	$0.60_{[0.49,0.69]}$	$0.67_{[0.54,0.77]}$	$0.72_{[0.55,0.92]}$	$0.78_{[0.66,0.94]}$	$0.76_{[0.59,0.91]}$
LR	–	$0.71_{[0.55,0.87]}$	$0.74_{[0.63,0.89]}$	$0.80_{[0.73,0.93]}$	$0.83_{[0.75,0.95]}$	$0.85_{[0.76,0.96]}$	$0.88_{[0.81,0.96]}$
TabPFN	–	$0.72_{[0.61,0.88]}$	$0.76_{[0.63,0.88]}$	$0.80_{[0.73,0.94]}$	$0.84_{[0.75,0.96]}$	$\mathbf{0.87}_{[\mathbf{0.79},\mathbf{0.97}]}$	$\mathbf{0.90}_{[\mathbf{0.83},\mathbf{0.98}]}$
LLaMA2-7B	$0.47_{[0.43,0.53]}$	$0.53_{[0.48,0.59]}$	$0.51_{[0.42,0.62]}$	$0.51_{[0.44,0.57]}$	$0.50_{[0.45,0.57]}$	$0.50_{[0.44,0.57]}$	$0.54_{[0.49,0.60]}$
LLaMA2-13B	$0.52_{[0.45,0.62]}$	$0.55_{[0.45,0.64]}$	$0.54_{[0.44,0.64]}$	$0.51_{[0.45,0.59]}$	$0.53_{[0.45,0.61]}$	$0.51_{[0.42,0.60]}$	$0.53_{[0.42,0.63]}$
Phi3-mini	$0.54_{[0.43,0.69]}$	$0.53_{[0.40,0.63]}$	$0.55_{[0.44,0.65]}$	$0.58_{[0.48,0.66]}$	$0.59_{[0.51,0.68]}$	$0.63_{[0.52,0.73]}$	$0.62_{[0.56,0.72]}$
Phi3-medium	$0.62_{[0.51,0.71]}$	$0.62_{[0.50,0.73]}$	$0.62_{[0.49,0.76]}$	$0.63_{[0.52,0.73]}$	$0.58_{[0.52,0.66]}$	$0.60_{[0.52,0.67]}$	$0.60_{[0.51,0.70]}$
LLaMA2-7B-GTL	$0.58_{[0.48,0.66]}$	$0.75_{[0.60,0.90]}$	$0.77_{[0.62,0.93]}$	$0.77_{[0.65,0.93]}$	$0.77_{[0.68,0.93]}$	$0.66_{[0.50,0.85]}$	$0.65_{[0.49,0.87]}$
LLaMA2-13B-GTL	$0.63_{[0.51,0.78]}$	$\mathbf{0.80}_{[\mathbf{0.68},\mathbf{0.93}]}$	$\mathbf{0.82}_{[\mathbf{0.72},\mathbf{0.96}]}$	$0.81_{[0.71,0.95]}$	$0.81_{[0.71,0.95]}$	$0.69_{[0.56,0.87]}$	$0.65_{[0.46,0.89]}$
LLaMA2-7B-GTL-S	$0.56_{[0.47,0.66]}$	$0.77_{[0.65,0.91]}$	$0.78_{[0.63,0.93]}$	$0.80_{[0.66,0.95]}$	$0.83_{[0.72,0.96]}$	$0.84_{[0.76,0.96]}$	$0.89_{[0.81,0.97]}$
LLaMA2-13B-GTL-S	$0.62_{[0.47,0.72]}$	$0.78_{[0.69,0.92]}$	$0.81_{[0.65,0.97]}$	$0.82_{[0.70,0.95]}$	$0.84_{[0.74,0.98]}$	$0.86_{[0.78,0.97]}$	$0.89_{[0.81,0.96]}$
Phi3-mini-GTL-S	$0.63_{[0.52,0.73]}$	$0.76_{[0.64,0.91]}$	$0.76_{[0.60,0.93]}$	$0.79_{[0.65,0.94]}$	$0.82_{[0.75,0.94]}$	$0.83_{[0.75,0.95]}$	$0.87_{[0.80,0.94]}$
Phi3-medium-GTL-S	$\mathbf{0.64}_{[\mathbf{0.56},\mathbf{0.78}]}$	$0.78_{[0.70,0.93]}$	$0.80_{[0.66,0.93]}$	$\mathbf{0.83}_{[\mathbf{0.70},\mathbf{0.94}]}$	$\mathbf{0.85}_{[\mathbf{0.78},\mathbf{0.97}]}$	$0.85_{[0.76,0.98]}$	$0.88_{[0.86,0.95]}$

Table 2: Averaged NMAE subscripted with 25% and 75% quantiles on held-out regression tasks.

Model	0	4	8	16	32	64	128
LightGBM	–	$0.40_{[0.22,0.50]}$	$0.38_{[0.22,0.50]}$	$0.37_{[0.22,0.48]}$	$0.37_{[0.23,0.47]}$	$0.31_{[0.13,0.36]}$	$0.24_{[0.06,0.29]}$
XGBoost	–	$0.44_{[0.20,0.47]}$	$0.39_{[0.14,0.46]}$	$0.36_{[0.12,0.39]}$	$0.34_{[0.08,0.34]}$	$0.29_{[0.07,0.42]}$	$0.23_{[0.05,0.31]}$
CatBoost	–	$0.38_{[0.18,0.47]}$	$0.30_{[0.14,0.37]}$	$0.27_{[0.12,0.33]}$	$0.25_{[0.11,0.30]}$	$0.43_{[0.08,0.31]}$	$0.29_{[0.04,0.27]}$
FTTransformer	–	$1.15_{[1.00,1.23]}$	$1.08_{[0.96,1.16]}$	$1.13_{[0.92,1.21]}$	$1.07_{[0.94,1.16]}$	$1.02_{[0.90,1.16]}$	$1.06_{[0.89,1.20]}$
XTab	–	$0.57_{[0.24,0.68]}$	$0.40_{[0.16,0.55]}$	$0.34_{[0.15,0.42]}$	$0.36_{[0.13,0.38]}$	$0.37_{[0.12,0.37]}$	$0.51_{[0.15,0.43]}$
LR	–	$0.41_{[0.15,0.51]}$	$0.37_{[0.09,0.56]}$	$0.33_{[0.05,0.51]}$	$0.37_{[0.05,0.56]}$	$0.55_{[0.09,0.39]}$	$0.26_{[0.03,0.28]}$
LLaMA2-7B	$1.37_{[0.81,1.00]}$	$1.26_{[1.00,1.00]}$	$1.32_{[1.00,1.00]}$	$1.14_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.24_{[1.00,1.00]}$	$1.21_{[1.00,1.00]}$
LLaMA2-13B	$1.60_{[0.66,1.05]}$	$1.56_{[0.51,1.00]}$	$1.59_{[0.89,1.00]}$	$1.55_{[0.75,1.00]}$	$1.03_{[0.87,1.00]}$	$1.20_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$
Phi3-mini	$1.03_{[1.00,1.00]}$	$0.94_{[1.00,1.00]}$	$0.91_{[1.00,1.00]}$	$0.91_{[1.00,1.00]}$	$0.90_{[1.00,1.00]}$	$0.93_{[1.00,1.00]}$	$0.93_{[1.00,1.00]}$
Phi3-medium	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$	$1.00_{[1.00,1.00]}$
LLaMA2-7B-GTL	$0.54_{[0.27,0.71]}$	$0.35_{[0.12,0.43]}$	$0.30_{[0.09,0.42]}$	$0.34_{[0.09,0.56]}$	$0.33_{[0.07,0.54]}$	$0.43_{[0.06,1.00]}$	$1.46_{[1.00,1.00]}$
LLaMA2-13B-GTL	$\mathbf{0.46}_{[\mathbf{0.17},\mathbf{0.68}]}$	$0.55_{[0.10,0.46]}$	$0.30_{[0.08,0.41]}$	$0.30_{[0.07,0.43]}$	$0.31_{[0.07,0.40]}$	$0.50_{[0.15,1.00]}$	$1.00_{[1.00,1.00]}$
LLaMA2-7B-GTL-S	$0.61_{[0.27,0.62]}$	$0.34_{[0.11,0.50]}$	$0.29_{[0.08,0.42]}$	$0.29_{[0.08,0.44]}$	$0.28_{[0.08,0.35]}$	$0.30_{[0.07,0.43]}$	$0.26_{[0.07,0.31]}$
LLaMA2-13B-GTL-S	$0.59_{[0.20,0.80]}$	$\mathbf{0.30}_{[\mathbf{0.08},\mathbf{0.33}]}$	$0.32_{[0.08,0.41]}$	$0.25_{[0.07,0.35]}$	$\mathbf{0.24}_{[\mathbf{0.07},\mathbf{0.27}]}$	$0.26_{[0.05,0.36]}$	$0.23_{[0.05,0.23]}$
Phi3-mini-GTL-S	$0.55_{[0.21,0.74]}$	$0.33_{[0.10,0.47]}$	$0.30_{[0.09,0.46]}$	$0.29_{[0.08,0.51]}$	$0.29_{[0.08,0.41]}$	$0.31_{[0.07,0.43]}$	$0.26_{[0.06,0.29]}$
Phi3-medium-GTL-S	$0.58_{[0.23,0.83]}$	$\mathbf{0.30}_{[\mathbf{0.07},\mathbf{0.40}]}$	$\mathbf{0.26}_{[\mathbf{0.07},\mathbf{0.33}]}$	$\mathbf{0.24}_{[\mathbf{0.07},\mathbf{0.30}]}$	$0.25_{[0.05,0.32]}$	$\mathbf{0.25}_{[\mathbf{0.07},\mathbf{0.32}]}$	$\mathbf{0.20}_{[\mathbf{0.06},\mathbf{0.24}]}$

References

[1] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[2] Winky KO Ho, Bo-Sin Tang, and Siu Wai Wong. Predicting property prices with machine learning algorithms. *Journal of Property Research*, 38(1):48–70, 2021.

[3] Marko Bohanec, Mirjana Kljajić Borštnar, and Marko Robnik-Šikonja. Explaining machine learning models in sales predictions. *Expert Systems with Applications*, 71:416–428, 2017.

[4] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *KDD*, 2016.

[5] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *NeurIPS*, 2017.

[6] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: unbiased boosting with categorical features. In *NeurIPS*, 2018.

[7] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *NeurIPS*, 2021.

[8] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 2022.

[9] Hangting Ye, Wei Fan, Xiaozhuang Song, Shun Zheng, He Zhao, Dan dan Guo, and Yi Chang. PTaRL: Prototype-based tabular representation learning via space calibration. In *ICLR*, 2024.

[10] Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. TabR: Tabular deep learning meets nearest neighbors. In *ICLR*, 2024.

[11] Zifeng Wang and Jimeng Sun. TransTab: Learning transferable tabular transformers across tables. In *NeurIPS*, 2022.

[12] Roman Levin, Valeriia Cherepanova, Avi Schwarzschild, Arpit Bansal, C. Bayan Bruss, Tom Goldstein, Andrew Gordon Wilson, and Micah Goldblum. Transfer learning with deep tabular models. In *ICLR*, 2023.

[13] Bingzhao Zhu, Xingjian Shi, Nick Erickson, Mu Li, George Karypis, and Mahsa Shoaran. XTab: Cross-table pretraining for tabular transformers. In *ICML*, 2023.

[14] Yazheng Yang, Yuqi Wang, Guang Liu, Ledell Wu, and Qi Liu. UniTabE: A universal pretraining protocol for tabular foundation model in data science. In *ICLR*, 2024.

[15] Jiahuan Yan, Bo Zheng, Hongxia Xu, Yiheng Zhu, Danny Chen, Jimeng Sun, Jian Wu, and Jintai Chen. Making pre-trained language models great on tabular prediction. In *ICLR*, 2024.

[16] Chao Ye, Guoshan Lu, Haobo Wang, Liyao Li, Sai Wu, Gang Chen, and Junbo Zhao. Towards cross-table masked pretraining for web data mining. In *WWW*, 2024.

[17] Xumeng Wen, Han Zhang, Shun Zheng, Wei Xu, and Jiang Bian. From supervised to generative: A novel paradigm for tabular deep learning with large language models. In *KDD*, 2024.

[18] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

[20] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[21] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[22] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[23] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[24] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 2015.

[26] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. TabTransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678*, 2020.

[27] Sercan Ö Arik and Tomas Pfister. TabNet: Attentive interpretable tabular learning. In *AAAI*, 2021.

[28] Liran Katzir, Gal Elidan, and Ran El-Yaniv. Net-DNF: Effective deep modeling of tabular data. In *ICLR*, 2021.

[29] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. VIME: Extending the success of self- and semi-supervised learning to tabular domain. In *NeurIPS*, 2020.

[30] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342*, 2021.

[31] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. SubTab: Subsetting features of tabular data for self-supervised representation learning. In *NeurIPS*, 2021.

[32] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. SCARF: Self-supervised contrastive learning using random feature corruption. In *ICLR*, 2022.

[33] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. TabLLM: Few-shot classification of tabular data with large language models. In *AISTATS*, 2023.

[34] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. LIFT: Language-interfaced fine-tuning for non-language machine learning tasks. In *NeurIPS*, 2022.

[35] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *ICLR*, 2023.

[36] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. Multitask prompted training enables zero-shot task generalization. In *ICLR*, 2022.

[37] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. In *ACL*, 2020.

[38] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

[39] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

A Appendix

A.1 Holdout dataset information

We collected 18 classification datasets and 18 regression datasets from Kaggle for holdout evaluation. Detailed information about these datasets is provided in Table 3.

Table 3: A summary of 36 holdout datasets for evaluation.

Index	Dataset	Tags	Task	# Rows	# Classes	# Num features	# Cat features	Label max	Label min	Feature max	Feature min
1	utrecht-housing-dataset (url)	housing	Reg.	100	-1	10	4	1.34e+06	440000	1.14e+06	1
2	student-marks-dataset (url)	education	Reg.	100	-1	2	0	55.299	5.609	8	0.096
3	higher-education-students-performance-evaluation (url)	research	Reg.	145	-1	4	26	7	0	5	1
4	cholesterol (url)	heart conditions	Reg.	303	-1	4	9	564	126	202	0
5	body-fat-extended-dataset (url)	exercise	Reg.	436	-1	13	2	47.5	0	164.72	0.75
6	amsterdam-house-price-prediction (url)	housing	Reg.	920	-1	4	2	5.95e+06	175000	623	1
7	alcohol-effects-on-study (url)	education	Reg.	1044	-1	5	17	20	0	75	0
8	airfoil-selfnoise-dataset (url)	earth and nature	Reg.	1503	-1	5	0	140.987	103.38	20000	0
9	employeedataset (url)	universities and colleges	Reg.	3000	-1	0	21	5	1	-	-
10	uncovering-trends-in-health-outcomes-and-socioec (url)	housing	Reg.	3047	-1	30	2	362.8	59.7	1.02e+07	0
11	abalone-age-prediction (url)	earth and nature	Reg.	3550	-1	7	3	29	1	565.1	0
12	dummy-advertising-and-sales-data (url)	business	Reg.	4566	-1	3	1	364.08	31.20	100	3.13e-05
13	gold-rate-history-in-tamilnadu-india (url)	business	Reg.	4971	-1	1	3	5687	768	5416	711
14	co2-emission-by-vehicles (url)	earth and nature	Reg.	7385	-1	6	5	522	96	69	0.9
15	melody-metrics-decoding-song-popularity (url)	music	Reg.	8020	-1	5	1	100	0	2022	1.67e-05
16	crop-yield-in-indian-states-dataset (url)	india	Reg.	19689	-1	6	3	21105	0	6.33e+09	0
17	cern-electron-collision-data (url)	earth and nature	Reg.	100000	-1	14	4	109.999	2.00	948.375	-840.987
18	car-prices-poland (url)	europe	Reg.	117927	-1	3	6	2.40e+06	500	2.80e+06	0
19	entrepreneurial-competency-in-university-students (url)	universities and colleges	Cls.	219	2	7	9	1	0	26	1
20	conversion-predictors-of-cis-to-multiple-sclerosis (url)	diseases	Cls.	273	2	4	14	1	0	77	0
21	breast-cancer-data (url)	health	Cls.	286	2	0	9	1	0	-	-
22	disease-symptoms-and-patient-profile-dataset (url)	medicine	Cls.	349	2	1	8	1	0	90	19
23	go-to-college-dataset (url)	universities and colleges	Cls.	1000	2	4	6	1	0	1.00e+07	20
24	telecom-churn-datasets (url)	business	Cls.	3333	2	15	4	1	0	395	0
25	stroke-prediction-dataset (url)	health	Cls.	5110	2	3	7	1	0	271.74	0.08
26	predicting-credit-card-customer-attrition-with-m (url)	business	Cls.	10127	2	14	5	1	0	34516	0
27	nasa-near-earth-objects-information (url)	earth and nature	Cls.	24000	2	5	7	1	0	631.90	0.00
28	lending-club-loan-preprocessed-dataset (url)	business	Cls.	396030	2	12	11	1	0	8.71e+06	0
29	bank-account-fraud-dataset-neurips-2022 (url)	finance	Cls.	1000000	2	19	12	1	0	16754.96	-177
30	2d-clustering-data (url)	clustering	Cls.	336	3	2	0	2	0	698.54	201.33
31	datasets-in-it-ops-applied-ai (url)	earth and nature	Cls.	1000	3	3	4	2	0	1	0
32	maternal-health-risk-data (url)	health	Cls.	1014	3	6	0	2	0	160	6
33	iris-dataset-extended (url)	earth and nature	Cls.	1200	3	19	1	2	0	299.9	-1.55
34	stellar-classification-dataset-sdss17 (url)	earth and nature	Cls.	100000	3	9	6	2	0	58932	-9999
35	video-games-rating-by-esrb (url)	video games	Cls.	2395	4	0	33	3	0	-	-
36	factors-affecting-children-anemia-level (url)	africa	Cls.	13136	4	4	11	3	0	218	1

A.2 Baselines

Our experiments include two categories of baselines: large language models (LLMs) and traditional tabular models. For the LLM baselines, we evaluate on the LLaMA2-7B and LLaMA2-13B checkpoints, as well as the Phi3-3.8B and Phi3-14B checkpoints. On the traditional tabular model side, we evaluate several approaches. LightGBM [5], XGBoost [4] and CatBoost [6] are gradient boosting methods known for their strong performance in tabular data tasks. Logistic Regression (LR) and Linear Regression (LR) are straightforward and efficient, often serving as baseline models in classification and regression tasks respectively. FTTransformers [7] employs transformer-based architectures for tabular data, providing a deep learning approach tailored to the specific challenges of structured data. TabPFN [35] combines Positional Feature-wise Networks with transformer-based architectures, offering another advanced deep learning model for tabular data. In order to optimize the results for these baseline methods, we use z-score normalization for numerical features and labels, and apply one-hot encoding for categorical features.