TabLLM: Few-shot Classification of Tabular Data with Large Language Models

Stefan Hegselmann[1,2] **Alejandro Buendia**[1] **Hunter Lang**[1] **Monica Agrawal**[1] **Xiaoyi Jiang**[2] **David Sontag**[1]
[1] MIT CSAIL [2] University of Münster

Abstract

We study the application of large language models to zero-shot and few-shot classification of *tabular data*. We prompt the large language model with a serialization of the tabular data to a natural-language string, together with a short description of the classification problem. In the few-shot setting, we fine-tune the large language model using some labeled examples. We evaluate several serialization methods including templates, table-to-text models, and large language models. Despite its simplicity, we find that this technique outperforms prior deep-learning-based tabular classification methods on several benchmark datasets. In most cases, even *zero*-shot classification obtains non-trivial performance, illustrating the method's ability to exploit prior knowledge encoded in large language models. Unlike many deep learning methods for tabular datasets, this approach is also competitive with strong traditional baselines like gradient-boosted trees, especially in the very-few-shot setting.

1 INTRODUCTION

Many real world applications generate *tabular data* as a natural byproduct of relational databases (Shwartz-Ziv and Armon, 2022). It is ubiquitous in domains ranging from healthcare to climate and finance (Sahakyan et al., 2021). Obtaining enough labeled data to train supervised learning algorithms for classification can be difficult. For example, in healthcare, there are 10,000 rare diseases (Haendel et al., 2020) affecting very few patients, which hampers the development of risk stratification models. Thus, we seek to develop methods that can exploit prior knowledge (e.g., from medical articles) to improve predictive performance in settings with a small number of training examples, i.e. the *few-shot* setting.

While deep learning has led to breakthroughs in computer vision and natural language processing, this success has not yet been extended to the tabular domain. For example, self-supervised deep learning methods have been introduced for tabular data (Yin et al., 2020; Arik and Pfister, 2021), but Grinsztajn et al. (2022) showed that these deep techniques still underperform ensembles of gradient boosted trees in the fully supervised setting. This disparity in performance can be attributed to the differences between tabular data and text or images; tabular data lacks locality, contains mixed data types, and the number of columns is usually fairly small compared to the number of features in text or image data (Borisov et al., 2022a).

Recently, large language models (LLMs) such as GPT-3, which are pre-trained on enormous corpora of text, have shown incredible performance on few-shot text classification and generation tasks (Brown et al., 2020; Sanh et al., 2022; Ouyang et al., 2022). These LLMs perform well on a variety of tasks and domains, including fact retrieval (Liu et al., 2021), mathematical reasoning (Wei et al., 2022), medical information extraction (Agrawal et al., 2022), and tabular data cleaning tasks (Narayan et al., 2022). Most importantly, because of all the knowledge encoded in their parameters, LLMs require little or no *labeled* training data to obtain this good performance.

In this work we introduce *TabLLM*, which is a general framework to leverage LLMs for few-shot *classification* of tabular data. We prompt the LLM with a serialization of a row to a natural-language representation and a short description of the classification problem. For risk stratification, for instance, this serialization could list relevant patient attributes and combine it with, "Will this patient be hospitalized?". We experiment with nine different serializations and the T0 language model of different sizes (Sanh et al., 2022). We use the parameter-efficient fine-tuning method T-Few (Liu et al., 2022) to update the LLM's parameters using some labeled examples. We also evaluate GPT-3 in the zero-shot setting (Brown et al., 2020). To the best of our knowledge, this is one of the widest evaluations of LLMs for zero- and few-shot tabular classification.

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

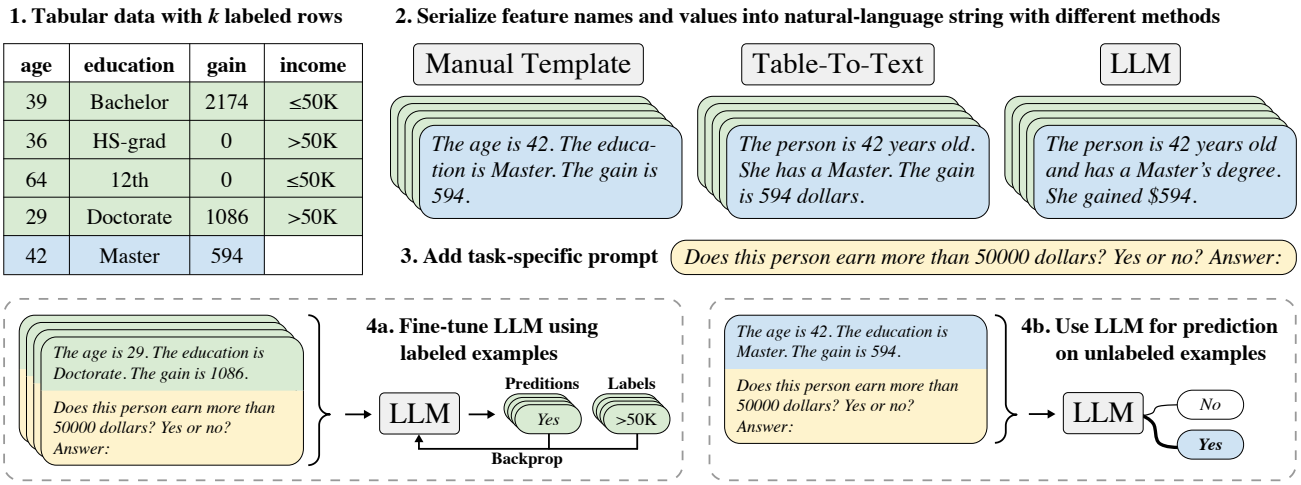

Figure 1: Overview of TabLLM. We first serialize the feature names and values into a natural language string. We evaluate different strategies. This string is then combined with a task-specific prompt. To get predictions, we obtain output probabilities from the LLM for each of a pre-specified set of verbalizer tokens (e.g., "Yes", "No"), which map to class labels (e.g., 1, -1). If $k > 0$, we use the k labeled examples to fine-tune the large language model using T-Few (Liu et al., 2022). Finally, we use the (possibly tuned) large language model to obtain predictions on unlabeled examples.

Despite its simplicity, we find that TabLLM outperforms prior deep-learning-based tabular classification methods on several benchmark datasets. By using information from the natural-language column names and feature values, it often enables effective *zero*-shot classification of tabular data. Unlike many deep learning methods on tabular data, this approach is also competitive with gradient-boosted tree baselines and outperforms them or is on par until 256 shots. In the very-few-shot setting it outperforms them by a considerable margin. The main contributions of this work are:

- We introduce TabLLM, a novel framework leveraging LLMs for data-efficient tabular classification
- We study nine serialization techniques and explore their performance across ten different datasets
- We show that TabLLM instantiated with a simple text serialization and the T0 LLM can outperform state-of-the-art neural models and tree ensembles in the zero- and few-shot setting
- We investigate the application of TabLLM to a large real-world healthcare claims dataset and introduce serialization methods that deal with many input features

2 RELATED WORK

2.1 Machine Learning on Tabular Data

Due to the success of deep learning in other domains, there have been many recent attempts at representation learning for tabular data. Self-supervised objectives have largely revolved around the prediction of masked cells, the identification or correction of corrupted cells, and contrastive losses over augmentations (Bahri et al., 2022; Somepalli et al., 2021; Yoon et al., 2020; Arik and Pfister, 2021; Huang et al., 2020). Additional efforts have included differentiable trees, which combine advantages of tree ensembles with gradient based optimization of neural networks (Kontschieder et al., 2015; Popov et al., 2020). However, several recent comprehensive reviews (Shwartz-Ziv and Armon, 2022; Borisov et al., 2022a; Grinsztajn et al., 2022) found that gradient-boosted tree ensembles like XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) systematically outperform these novel deep learning architectures, even with proper fine-tuning and regularization (Kadra et al., 2021). Levin et al. (2022) found utility in transfer learning in the semi-supervised setting, but required a set of additional supervised tasks on the same table, which can be a nontrivial limitation. They investigate few-shot classification for medical diagnosis using 4 to 200 labeled examples, but do not exploit the power of large pre-trained models, as we do in this work. Hollmann et al. (2022) recently introduced TabPFN, a Bayesian neural network pre-trained on synthetic tabular data, outperforming gradient boosted trees in a comprehensive evaluation.

2.2 Large Language Models for Tabular Data

Another approach has been to leverage the natural language capabilities of language models. Yin et al. (2020) use a language model for semantic parsing of natural language queries over tabular data. Li et al. (2020) investigate the ability of language models to perform entity matching on tabular data, i.e. determining if two rows refer to the same object. Harari and Katz (2022) study data enrichment by linking each table row with additional unstructured text (e.g., from Wikipedia) from which they generated addi-

tional features using a language model. However, this setup requires named entities (e.g., celebrities, universities, etc.), which is quite limiting. Bertsimas et al. (2022) studied two healthcare datasets and used a language model to generate feature embeddings, which they fed into classifiers like gradient boosted trees. All these studies use a BERT-style language model (Devlin et al., 2019). Narayan et al. (2022) recently assessed in-context learning with the autoregressive language model GPT-3 for tabular data cleaning tasks. They found that it often outperforms state-of-the-art approaches with ten labeled examples. Borisov et al. (2022b) introduced an LLM-agnostic method to generate realistic tabular data and found that it achieved better results than existing approaches. In contrast, here we study classification tasks of tabular data and investigate parameter-efficient fine-tuning of LLMs.

To use an LLM for tabular data, the table must be serialized into a natural text representation. All aforementioned works relied on simple list or sentence serializations; Yin et al. (2020) also included the column data type in the serialized string. Only Bertsimas et al. (2022) studied different serialization variants, but this was in a different context of deriving feature embeddings from BERT-style language models. The LIFT method introduced by Dinh et al. (2022) comes closest to our work. The authors evaluated the capabilities of fine-tuned GPT-3 and GPT-J models for regression and classification on synthetic, tabular, and vision data. They also studied the sample efficiency and considered different static serialization templates assessing the effect of including column names in the input. In this work, we focus on the publicly available T0 model and perform a broader analysis of nine serialization techniques including automatic approaches and ablations evaluating the importance of feature values. Particularly, we are interested in leveraging prior knowledge encoded in LLMs and we do a more fine-grained analysis of the sample efficiency including zero-shot experiments on ten different datasets.

3 METHODS

3.1 TabLLM for Tabular Data Classification

Problem Formalization. Suppose we have a tabular dataset with n rows and d columns or features. We can formalize this as $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, where each $\mathbf{x}_i$ is a d-dimensional feature vector. Since we consider classification, $y_i \in C$ for a set of classes C. We define the column names or feature names as $F = \{f_1, ..., f_d\}$. We assume the f_i's are natural-language strings such as "age" or "education" (see Figure 1). For our k-shot classification experiments, we only use a subset D_k of size k—sampled from D with replacement—for fine-tuning or training.

Serialization of Tabular Data. To use an LLM for tabular data, the table must be transformed into a natural text representation. Typically, when prompting an LLM, there is a template used to both serialize the inputs into one natural-language string, and to provide the prompt itself (e.g., the string *"Does this person make more than 50,000 dollars? Yes or no?"*), which is usually located after the serialized input. In this work, we break these pieces up into a *serialization* and a *prompt*. We define a function $\texttt{serialize}(F, \mathbf{x})$ that takes the column names F and feature values $\mathbf{x}$ for a row as inputs and creates a textual representation of the input. Combining this serialization with a task-specific prompt p will then form the LLM input $(\texttt{serialize}(F, \mathbf{x}), p)$. This is illustrated in Figure 1. We primarily study the serialization, since that is the biggest difference compared to existing applications of prompting. Previous work has usually considered a simple concatenation of feature names and values as a serialization of tabular data (Li et al., 2020; Narayan et al., 2022). In our work, this function can be arbitrarily complex. For instance, we explore serializations that include (i) incorporating another LLM and (ii) employing feature selection as a substep.

Large Language Models For Classification TabLLM can be used with different LLMs that generate text based on a natural-language input. Let $\texttt{LLM}$ be an LLM with vocabulary V. Then, $\texttt{LLM}((\texttt{serialize}(F, \mathbf{x}), p)) \in V^*$ is the prompted output of the LLM. In our few-shot setting, $\{(\texttt{serialize}(F, \mathbf{x}), p) \mid (\mathbf{x}, y) \in D_k\}$ can be used as training examples for fine-tuning the LLM. The LLM generates text in the vocabulary space V^* that has to be mapped to a valid class in C. Several approaches already exist for this problem. For example, the verbalizer (Schick and Schütze, 2021) defines a mapping between LLM output tokens and the discrete label space. Verbalizers can be manually specified or automatically learned; see Cui et al. (2022) for an overview of different verbalizer-learning approaches. In this work, we assume for simplicity that the verbalizer mapping is manually specified (see `answer_choices` in the templates in Sec. 8 in the Supplement).

3.2 Our Instantiation of TabLLM

Serialization Approaches for TabLLM. The performance of LLMs is very sensitive to the precise details of the natural-language input (Zhao et al., 2021; Webson and Pavlick, 2022). In this work, we focus on the serialization of the tabular data. For the prompt, we use a simple description of the classification task and perform no further prompt engineering. We study nine different serialization formats varying in complexity. All serialization methods require minimal human effort to apply to new classification tasks. We evaluate several methods that generate natural text to create inputs that are closer to the training distribution of the LLM, thereby improving zero and very-few-shot performance. Additional details and examples for the serializations are given in Sec. 1.2.1 and 9 in the Supplement.

- **List Template**: A list of column names and feature values. We fixed an arbitrary ordering of the columns.
- **Text Template**: An textual enumeration of all features as "The *column name* is *value*." (see Figure 1).
- **Table-To-Text**: We use an LLM fine-tuned on a table-to-text generation task from HuggingFace (`Narrativaai/bloom-560m-finetuned-totto-table-to-text`). To ensure that the serialization includes all data we hand each column-value tuple to the model separately and concatenate the outputs.
- **Text T0**: We use the LLM T0 with 11B parameters (`bigscience/T0pp`) (Sanh et al., 2022). We split up a row into pairs of two column-value tuples. We send them to LLM separately with the prompt "Write this information as a sentence:" and combine the outputs.
- **Text GPT-3**: We use GPT-3 (engine *text-davinci-002*) accessible through an API (Ouyang et al., 2022). GPT-3 was able to serialize all features at once, so we use a list of all features with the prompt "Rewrite all list items in the input as a natural text." as input. We guide the output with "The {person, car, patient} is".

We consider the following serializations as ablations:

- **List Only Values**: *List Template* for feature values only. We want to evaluate whether column names aid the classification performance.
- **List Permuted Names**: *List Template* with permuted column names. Hence, the wrong column name is associated with each feature value. The permutation is the same across all examples. We perform this ablation to study the relevance of the correct association between column names and feature values.
- **List Permuted Values**: *List Template* with consistently permuted values across all examples. We generate one permutation for each column and apply this mapping to all column values. For continuous values, we use ten uniform bins. This tests whether the LLM uses the fine-grained information encoded by the feature *values* for zero-shot and few-shot classification.
- **List Short**: *List Template* with at most ten features. We only consider this for the healthcare dataset where the number of features exceeds the input limit of the LLM. We want to study the effect of less information.

Large Language Models for TabLLM Another crucial component of TabLLM is the LLM. TabLLM is both agnostic to the LLM and the specific fine-tuning method that is used. We only consider a single LLM for most of our experiments. We employ the T0 encoder-decoder model with 11 billion parameters as the LLM for TabLLM (Sanh et al., 2022). It was trained on a large variety of task-specific prompts, making it a suitable candidate for our experiments (Sanh et al., 2022). This model has a token limit of 1024, which roughly corresponds to 400 words. We also evaluate the effect of a smaller version of the T0 model (T0 3B). We fine-tuned on the few-shot data $\mathcal{D}_k$ using the recent T-Few recipe, which outperforms other parameter-efficient tuning methods such as soft prompt tuning (Liu et al., 2022). In addition, we perform zero-shot experiments with the LLM GPT-3 (engine *text-davinci-002*) (Ouyang et al., 2022).

4 EXPERIMENTAL SETUP

4.1 Datasets

We studied TabLLM in two experimental settings. First, we considered nine medium-sized tabular datasets for binary and multi-class classification. We systematically identified datasets from Kadra et al. (2021), Grinsztajn et al. (2022), and Borisov et al. (2022a). We included datasets with at most 50,000 rows to keep the fine-tuning costs manageable and at most 30 columns to stay within T0's token limit. We also required textual feature names to make the serializations more meaningful and we excluded datasets with derived feature values (e.g., mean pixel values). This lead to inclusion of **Bank** (45,211 rows, 16 feats), **Blood** (748, 4), **California** (20,640, 8), **Car** (1,728, 8), **Credit-g** (1,000, 20), **Income** (48,842, 14), and **Jungle** (44,819, 6). We added two additional datasets from Kaggle that fulfilled our inclusion criteria: **Diabetes** (768, 8) and **Heart** (918, 11). Second, we evaluated TabLLM for risk stratification on three binary classification tasks, following prior work by Kodialam et al. (2021) and similarly using a de-identified health claims dataset from a U.S. health insurer. We predicted the end-of-life (**EoL**) of all patients older than 70 years, which can be used to inform care in a palliative setting (Avati et al., 2018). We also considered the need for any surgical procedure (**Surgery**) and the likelihood of hospitalization (**LoH**), which can help with determining health care needs and estimating future costs. Additional details on all datasets can be found in Sec. 1 in the Supplement. We release the code for our experiments on Github.[1]

4.2 LLM and Fine-tuning

We used the HuggingFace implementation of the T0 model (`bigscience/{T0pp,T0_3B}`). Prompts for the LLM were designed following Sanh et al. (2022) using the PromptSource framework (Bach et al., 2022). Each class in our classification tasks was manually encoded in a textual response, e.g., "Yes" and "No" for true and false (Sanh et al., 2022). The prediction probability for each class corresponds to the probability of the LLM generating its token sequence normalized across all classes. All templates used in this work are given in Sec. 8 in the Supplement.

[1] `https://github.com/clinicalml/TabLLM`

For fine-tuning, we adopted the default hyperparameters of the T-Few method without any additional parameter tuning (Liu et al., 2022). The authors used a setup of $k = 32$ shots and 1,000 training steps for most of their experiments, which corresponds to 31.25 epochs. Hence, we fixed 30 training epochs for all few-shot experiments on the public tabular datasets. We used 20% of the data as a test set. For the large healthcare claims dataset, we used 10 epochs for up to 256 shots and 3 epochs for 1,024, 4,096 and 16,384 to reduce the runtime and prevent overfitting for many training examples. We used a test set of 10,000 examples for the three healthcare tasks. All experiments were evaluated with the area under the receiver operating characteristic curve (AUC). We used macro-AUC one-versus-rest for the multiclass setting. Estimates for the runtime are given in Sec. 2 in the Supplement.

4.3 Baseline Models

We compared TabLLM to several baselines. For the simplest baseline, we used a logistic regression (LR) model. Since previous work showed the superiority of gradient boosted tree ensembles (Borisov et al., 2022a), we included the most common models XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017). We also evaluated several state-of-the-art deep learning baselines. TabNet is a widely used neural model for tabular data that uses attention over columns (Arik and Pfister, 2021). SAINT is a more recent approach that uses attention over rows and columns (Somepalli et al., 2021). SAINT performed best in a comprehensive review on tabular data (Borisov et al., 2022a). NODE is a differentiable tree ensemble method that performed best in the evaluation of Shwartz-Ziv and Armon (2022). Lastly, we include TabPFN, a Bayesian neural network that was pre-trained on synthetic tabular data (Hollmann et al., 2022). In contrast to TabLLM, we performed hyperparameter tuning for all baselines except TabPFN (see Sec. 3 in the Supplement), which requires no tuning by design. We adopted the parameter ranges from previous reviews (Borisov et al., 2022a; Grinsztajn et al., 2022). Since no validation set exists in the few-shot setting, we used 4-fold cross validation on the k-shots. In particular, we *did not* use a large validation set for hyperparameter tuning, unlike some few-shot learning works as highlighted by Perez et al. (2021). We encoded categorical values as one-hot vectors. We also tested ordinal encoding for LR, XGBoost, LightGBM, and TabPFN, but it showed worse results (see Table 12, 13, and 14 in the Supplement). In addition, we give results for GPT-3 (`text-davinci-002`) without fine-tuning, i.e. in the zero-shot setting using the *Text Template* serialization.

For the three health claims tasks, we used the same experimental setup for the baselines. However, we only included LR and LightGBM due to runtime limitations. Following Kodialam et al. (2021), each patient's input was a one-hot encoded vector. For each medical concept, there were three indicator variables of whether that concept occurred within 30 days, 1 year, and anytime before prediction time.

4.4 Serializations

For the public datasets, some column names and feature values were manually mapped to human-readable forms, based on the provided documentation. For instance, for the **Income** dataset, the feature name *hours_per_week* was mapped to *work hours per week* and the feature value *private* for working class was mapped to *private sector employee*. Numerical values were not changed.

Serialization was more complex for the healthcare claims data. Each patient record is a time series of visits, with each visit consisting of a list of medical conditions and procedures. We only considered the manual serializations *List Template* and *Text Template*. We tried to mimic the style of a medical professional to tap potential prior knowledge of the LLM. To this end, the serialization starts with an intro sentence containing the patient's gender, age, and race. It then describes each visit, stating its date, the type of doctor the patient saw (e.g., dermatology) if an outpatient visit or length of hospitalization if an inpatient visit, the primary complaint of the associated visit, and procedures performed. Since there are no feature values in this dataset, we omit *List Only Values* and *List Permuted Values*. We also performed experiments for concept selection and different names for the medical concepts. Details for these additional experiments and examples of the serializations are given in Sec. 1.2.2, 1.2.3, and 9 in the Supplement.

5 RESULTS

5.1 Effects of serialization

Figure 2 shows the performance of different serialization methods for TabLLM averaged over the nine public datasets. The *Text Template* serialization performed very well across all experiments. In the zero-shot setting, the *Text Template* showed improvements over *List Template*, indicating the benefit of a serialization that is closer to the training distribution of T0. However, these differences already vanished for 8 training examples. Hence, very few training examples might already suffice to adjust for different templates. This suggests that sophisticated serializations might be unnecessary when some training data exists.

Using LLMs for serialization showed mixed results. The ordering is according to the complexity of the LLM used for serialization. GPT-3 has 175B, T0 11B, and the BLOOM table-to-text model 0.56B parameters. Different reasons might be responsible for the worse performance overall. The models tended to hallucinate information for some examples, leading to biased predictions of TabLLM.

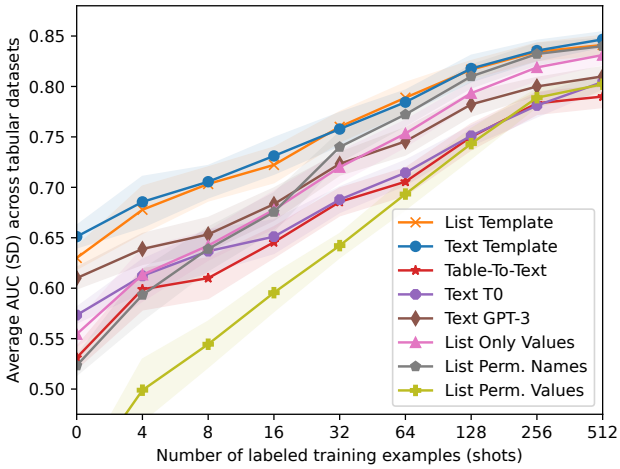

Figure 2: Average AUC and SD of different serializations across nine public datasets. *Text Template* performs best for zero and few training examples. For many examples, the performance of different serializations converges.

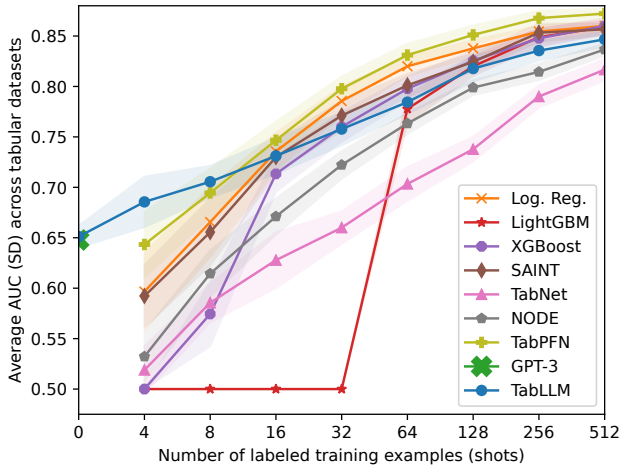

Figure 3: Average AUC and SD of TabLLM versus all baseline models across nine public datasets. TabLLM outperforms all baselines for zero and very few training examples. TabPFN is the strongest baseline.

For instance, GPT-3 added "this car is a good choice" or added entirely new data to some examples (see Sec. 9 in the Supplement). Also, the LLMs are not completely faithful at including all features, even though we tried to enforce it in our experiments. This could explain that none of the LLM serializations reaches the same performance as the template serializations, even for many training examples.

Using only feature *values* had a poor performance for zero and very few shots, but the performance equalized with more training examples. The same applies to the list serialization with permuted feature names. This indicates that if enough training examples are available, the serialization approach does not matter, but that TabLLM relies on information from the feature names in the zero-shot and few-shot regime, and also relies on the association of the names with the correct values. The discrepancy for zero and very few shots was even stronger for *List Permuted Values*, which suggests that TabLLM relies more on the correct values than feature names. Again, the performance equalized for more examples showing the ability of TabLLM to learn new associations if enough training data is available. Using the smaller T0 3B model showed a slightly decreased performance (see Table 12, 13, and 14 in the Supplement).

For the healthcare claims dataset, we found that the *List Template* slightly outperformed the *Text Template* serialization (see Table 15 in the Supplement). This was consistent across tasks. The *List Short* serialization only performed slightly worse. The evaluation of different concept selection strategies showed that choosing the most frequent conditions per patient performed best. We found no considerable performance difference for different concept names.

From here onwards, we show results for TabLLM using the *Text Template* serialization for the public datasets. For the healthcare claims dataset, we use the *List Template* serialization and select the most frequent conditions. Results for all (dataset, serialization) combinations (Table 12, 13, and 14) and the additional experiments on the healthcare dataset (Table 5 and 7) can be found in the Supplement.

5.2 Public Tabular Datasets

Figure 3 shows the averaged results for TabLLM using the best serialization (*Text Template*) versus all baseline models. Table 1 contains the detailed results for TabLLM, TabPFN, and XGBoost. TabLLM showed a similar behavior across datasets. It achieved nontrivial zero-shot performance for all tasks except on **Credit-g** and **Heart**. For **Heart** this might be due to the dataset's inclusion criteria requiring eligibility for a heart procedure biasing the prediction. In all cases, TabLLM's performance improved with a higher number of shots. In the zero-shot setting, TabLLM was on par with GPT-3 even though GPT-3 is a much larger model than T0 (175B vs. 11B parameters). TabPFN consistently outperformed the other baseline models across all numbers of training examples. TabPFN reached TabLLM's performance with 4 to 256 (**Income**) training examples. LR was the second-best baseline often beating the tree models, which might be due to our extensive parameter tuning (see Sec. 4 in the Supplement). TabLLM outperformed or was on par with the tree ensemble baselines until 256 training examples for all datasets except **Calhousing** and **Jungle**. For fewer shots, it often outperformed them by a large margin. XGBoost performed relatively poorly for few shots, which was probably due to overfitting on the small training and validation sets (as described in the previous section, we do *not* use large validation sets for hyperparameter tuning to ensure the results are truly few-shot). TabLLM outperformed the neural baselines SAINT, NODE, and TabNet in many settings. It also

Table 1: Test AUC performance of TabLLM, the best tree ensemble model (XGBoost), and the best baseline (TabPFN) on the public tabular datasets. Each column reports the performance for k training examples. TabLLM (T0 + *Text Template*) outperforms XGBoost and TabPFN in the very-few-shot regime. Standard deviations are given across five random seeds.

		Number of Shots									
Dataset	**Method**	**0**	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**all**
Bank	XGBoost	—	$0.50_{.00}$	$0.56_{.09}$	$0.68_{.04}$	$0.76_{.03}$	$\mathbf{0.83}_{.02}$	$0.85_{.03}$	$0.88_{.01}$	$0.90_{.01}$	$\mathbf{0.94}_{.00}$
	TabPFN	—	$0.59_{.14}$	$\mathbf{0.66}_{.08}$	$\mathbf{0.69}_{.02}$	$0.76_{.03}$	$0.82_{.03}$	$\mathbf{0.86}_{.02}$	$\mathbf{0.89}_{.00}$	$0.90_{.00}$	$0.91_{.00}$
	TabLLM	$\mathbf{0.63}_{.01}$	$0.59_{.10}$	$0.64_{.05}$	$0.65_{.05}$	$0.64_{.06}$	$0.69_{.03}$	$0.82_{.05}$	$0.87_{.01}$	$0.88_{.01}$	0.92 †
Blood	XGBoost	—	$0.50_{.00}$	$0.58_{.07}$	$0.66_{.04}$	$0.67_{.06}$	$0.68_{.05}$	$0.71_{.06}$	$0.70_{.07}$	$0.67_{.06}$	$0.71_{.04}$
	TabPFN	—	$0.52_{.08}$	$0.64_{.04}$	$\mathbf{0.67}_{.01}$	$\mathbf{0.70}_{.04}$	$\mathbf{0.73}_{.04}$	$\mathbf{0.75}_{.04}$	$\mathbf{0.76}_{.04}$	$\mathbf{0.76}_{.03}$	$\mathbf{0.74}_{.03}$
	TabLLM	$\mathbf{0.61}_{.04}$	$\mathbf{0.58}_{.09}$	$\mathbf{0.66}_{.03}$	$0.66_{.07}$	$0.68_{.04}$	$0.68_{.04}$	$0.68_{.06}$	$0.70_{.08}$	$0.68_{.04}$	$0.70_{.04}$
Calhousing	XGBoost	—	$0.50_{.00}$	$0.62_{.10}$	$0.74_{.03}$	$0.79_{.04}$	$0.82_{.04}$	$0.87_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$\mathbf{0.97}_{.00}$
	TabPFN	—	$0.63_{.13}$	$\mathbf{0.63}_{.11}$	$\mathbf{0.80}_{.03}$	$\mathbf{0.85}_{.03}$	$\mathbf{0.89}_{.01}$	$\mathbf{0.91}_{.01}$	$\mathbf{0.92}_{.00}$	$\mathbf{0.93}_{.00}$	$0.94_{.00}$
	TabLLM	$\mathbf{0.61}_{.01}$	$0.63_{.05}$	$0.60_{.07}$	$0.70_{.08}$	$0.77_{.08}$	$0.77_{.04}$	$0.81_{.02}$	$0.83_{.01}$	$0.86_{.02}$	$0.95_{.00}$
Car	XGBoost	—	$0.50_{.00}$	$0.59_{.04}$	$0.70_{.08}$	$0.82_{.03}$	$0.91_{.02}$	$0.95_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
	TabPFN	—	$0.64_{.06}$	$0.75_{.05}$	$\mathbf{0.87}_{.04}$	$\mathbf{0.92}_{.02}$	$\mathbf{0.97}_{.00}$	$\mathbf{0.99}_{.01}$	$\mathbf{1.00}_{.00}$	$1.00_{.00}$	$1.00_{.00}$
	TabLLM	$\mathbf{0.82}_{.02}$	$\mathbf{0.83}_{.03}$	$\mathbf{0.85}_{.03}$	$0.86_{.03}$	$0.91_{.02}$	$0.96_{.02}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$	$1.00_{.00}$
Credit-g	XGBoost	—	$0.50_{.00}$	$0.51_{.07}$	$0.59_{.05}$	$0.66_{.03}$	$0.67_{.06}$	$0.68_{.02}$	$0.73_{.02}$	$0.75_{.03}$	$\mathbf{0.78}_{.04}$
	TabPFN	—	$0.58_{.08}$	$0.59_{.03}$	$0.64_{.06}$	$0.69_{.07}$	$0.70_{.07}$	$\mathbf{0.72}_{.06}$	$\mathbf{0.75}_{.04}$	$0.75_{.02}$	$0.75_{.03}$
	TabLLM	$\mathbf{0.53}_{.05}$	$\mathbf{0.69}_{.04}$	$\mathbf{0.66}_{.04}$	$\mathbf{0.66}_{.05}$	$\mathbf{0.72}_{.06}$	$0.70_{.07}$	$0.71_{.07}$	$0.72_{.03}$	$0.72_{.02}$	$0.70_{.02}$
Diabetes	XGBoost	—	$0.50_{.00}$	$0.59_{.16}$	$\mathbf{0.72}_{.07}$	$0.69_{.08}$	$0.73_{.05}$	$0.78_{.05}$	$0.80_{.03}$	$0.80_{.01}$	$\mathbf{0.84}_{.03}$
	TabPFN	—	$0.61_{.13}$	$\mathbf{0.67}_{.11}$	$0.71_{.07}$	$\mathbf{0.77}_{.03}$	$\mathbf{0.82}_{.03}$	$\mathbf{0.83}_{.03}$	$\mathbf{0.83}_{.03}$	$\mathbf{0.81}_{.02}$	$0.81_{.03}$
	TabLLM	$\mathbf{0.68}_{.06}$	$0.61_{.09}$	$0.63_{.08}$	$0.69_{.07}$	$0.68_{.04}$	$0.73_{.03}$	$0.79_{.04}$	$0.78_{.02}$	$0.78_{.04}$	$0.80_{.04}$
Heart	XGBoost	—	$0.50_{.00}$	$0.55_{.14}$	$0.84_{.07}$	$0.88_{.04}$	$0.91_{.01}$	$0.91_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.94_{.01}$
	TabPFN	—	$\mathbf{0.84}_{.06}$	$\mathbf{0.88}_{.05}$	$0.87_{.06}$	$\mathbf{0.91}_{.02}$	$\mathbf{0.92}_{.02}$	$\mathbf{0.92}_{.02}$	$0.92_{.01}$	$0.92_{.02}$	$0.92_{.02}$
	TabLLM	$\mathbf{0.54}_{.04}$	$0.76_{.14}$	$0.83_{.05}$	$0.87_{.04}$	$0.87_{.06}$	$0.91_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.92_{.01}$	$0.94_{.01}$
Income	XGBoost	—	$0.50_{.00}$	$0.59_{.06}$	$0.77_{.02}$	$0.79_{.03}$	$0.82_{.02}$	$0.84_{.01}$	$0.87_{.01}$	$0.88_{.00}$	$\mathbf{0.93}_{.00}$
	TabPFN	—	$0.73_{.08}$	$0.71_{.09}$	$0.76_{.09}$	$0.80_{.04}$	$0.82_{.04}$	$0.84_{.01}$	$0.86_{.01}$	$0.87_{.01}$	$0.89_{.00}$
	TabLLM	$\mathbf{0.84}_{.00}$	$\mathbf{0.84}_{.01}$	$\mathbf{0.84}_{.02}$	$\mathbf{0.84}_{.04}$	$\mathbf{0.84}_{.01}$	$\mathbf{0.84}_{.02}$	$\mathbf{0.86}_{.01}$	$0.87_{.00}$	$\mathbf{0.89}_{.01}$	$0.92_{.00}$
Jungle	XGBoost	—	$0.50_{.00}$	$0.58_{.07}$	$\mathbf{0.72}_{.05}$	$0.78_{.03}$	$0.81_{.02}$	$0.84_{.02}$	$0.87_{.01}$	$0.91_{.01}$	$0.98_{.00}$
	TabPFN	—	$\mathbf{0.65}_{.08}$	$\mathbf{0.72}_{.04}$	$0.71_{.07}$	$0.78_{.02}$	$0.81_{.01}$	$0.84_{.01}$	$\mathbf{0.88}_{.01}$	$0.91_{.00}$	$0.93_{.00}$
	TabLLM	$\mathbf{0.60}_{.00}$	$0.64_{.01}$	$0.64_{.02}$	$0.65_{.03}$	$0.71_{.02}$	$0.78_{.02}$	$0.81_{.02}$	$0.84_{.01}$	$0.89_{.01}$	**1.00** †

† These experiments were only performed for a single run due to runtime limitations of TabLLM on the full dataset.

Table 2: Five highest and lowest weighted features for zero-shot TabLLM and logistic regression (LR) trained on all data for **Income**. Both models show very similar trends for important features.

Feature	**TabLLM**		**LR**	
	rank	weight	rank	weight
capital_gain	1	5.310	2	2.393
education_Masters	2	4.623	6	1.455
education_Doctorate	3	3.410	4	2.066
education_Bachelors	4	2.995	7	1.135
education_Prof-school	5	2.949	5	1.900
occupation_Priv-house-serv	102	-2.840	105	-1.909
education_12th	103	-3.178	79	-0.480
education_Preschool	104	-3.520	106	-2.385
occupation_Farming-fishing	105	-3.853	98	-0.982
workclass_Without-pay	106	-4.423	69	-0.174

was on par or very close to the best baseline models on the full datasets, indicating that there is little performance lost due to the serialization and the choice of model family.

Introspecting TabLLM—What Prior Knowledge Does it Use? Given the strong zero-shot performance of TabLLM on the **Income** dataset, we next sought to understand which features it based its predictions on in order to shed light on the prior knowledge used by the LLM. To determine the feature importance for TabLLM, we fit a LR model to the zero-shot prediction using the original features as covariates as described in Sec. 6 in the Supplement. Highly weighted features (see Table 2) for zero-shot TabLLM include the individual's occupation (with e.g., 'Farming-fishing' having a large negative weight), highest education level ('Masters' and 'Doctorate' have positive weights; 'Preschool' grade has a negative weight), and workclass ('Without-pay' has a negative weight). TabLLM also seems to be able to correctly interpret the numerically encoded capital gain value. For comparison, we also show the feature weights for a LR model trained on all data. We see a strong concordance between both models; TabLLM's top five features are all among the top seven of the LR model. However, TabLLM scores the highest education

Table 3: Test AUC on the healthcare claims dataset. TabLLM outperforms logistic regression (LR) for up to 64 and LightGBM for up 256 training examples on End of Life (**EoL**). Standard deviations are given across five random seeds.

		Number of Shots							
Dataset	**Method**	**0**	**16**	**64**	**256**	**1,024**	**4,096**	**16,384**	**all**
EoL	LR	—	$0.65_{.07}$	$0.77_{.02}$	$\mathbf{0.80}_{.02}$	$\mathbf{0.83}_{.01}$	$\mathbf{0.83}_{.01}$	$\mathbf{0.84}_{.01}$	$\mathbf{0.84}_{.01}$
	LightGBM	—	$0.50_{.00}$	$0.71_{.01}$	$0.76_{.02}$	$0.80_{.01}$	$0.82_{.01}$	$0.83_{.01}$	0.82 †
	TabLLM	**0.70**	**0.74**	**0.78**	0.78	0.79	0.81	0.81	—
Surgery	LR	—	$0.72_{.04}$	$\mathbf{0.75}_{.05}$	$0.77_{.01}$	$0.79_{.01}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
	LightGBM	—	$0.50_{.00}$	$0.73_{.02}$	$0.77_{.01}$	$0.79_{.01}$	$0.80_{.00}$	$0.81_{.01}$	**0.82** †
	TabLLM	**0.67**	**0.73**	0.72	0.73	0.75	0.78	0.79	—
LoH	LR	—	$0.72_{.04}$	$\mathbf{0.76}_{.03}$	$\mathbf{0.80}_{.01}$	$\mathbf{0.82}_{.01}$	$0.83_{.01}$	$0.83_{.01}$	$0.84_{.01}$
	LightGBM	—	$0.50_{.00}$	$0.72_{.02}$	$0.76_{.03}$	$0.81_{.01}$	$0.83_{.00}$	$0.83_{.01}$	**0.85** †
	TabLLM	**0.71**	**0.73**	0.73	0.76	0.78	0.81	0.82	—

† These experiments were only performed for a single run due to runtime limitations on the full dataset.

Table 4: Five highest and lowest weighted features for zero-shot TabLLM for **EoL** and their relative risk (RR) with confidence intervals (CI). The top five features show a significant increase of the relative risk.

Feature	TabLLM	RR (95% CI)
atrial fibrillation	0.633	2.72 (2.51-2.95)
atherosclerosis of coronary art...	0.530	2.10 (1.94-2.27)
atherosclerosis of aorta	0.473	1.99 (1.81-2.19)
exudative age-related macular d...	0.452	2.38 (2.06-2.75)
sex_male	0.442	1.23 (1.14-1.33)
open angle with borderline intr...	-0.338	1.20 (1.03-1.40)
primary localized osteoarthrosi...	-0.366	1.08 (0.82-1.43)
localized, primary osteoarthritis	-0.393	1.23 (1.07-1.40)
sex_female	-0.441	0.81 (0.75-0.88)
open-angle glaucoma - borderline	-0.495	0.97 (0.85-1.10)

degrees in the opposite order. Table 16 in the Supplement shows the importance of all 106 features.

5.3 Large Healthcare Claims Dataset

Table 3 shows the results for TabLLM with the *List Template* serialization on **EoL**, **Surgery**, and **LoH**, the three prediction tasks for the healthcare claims dataset. TabLLM showed very considerable zero-shot performance, ranging from 0.67 AUC for **Surgery** to 0.71 for **LoH**. The performance improves with higher number of training examples. However, the performance jumps happen at different steps and to a different extent. TabLLM outperformed LR for up to 16 (**Surgery** and **LoH**) to 64 (**EoL**) training examples and LightGBM for up to 64 (**LoH**) and 256 (**EoL**) examples. For more examples, LR and LightGBM performed slightly better. This could suggest that the information lost from our concept selection procedure, needed because of the token limits of the LLM, eventually starts costing TabLLM performance. We also evaluated TabLLM and LR in an unbalanced setting (see Table 15 in the Supplement). In this case, TabLLM outperforms LR up to 64 training examples on all datasets emphasizing its utility in a real world setting with limited access to labeled data.

Introspecting TabLLM—What Prior Knowledge Does it Use? We also performed a feature analysis to study the strong zero-shot performance on **EoL**. However, we did not compare to a LR model trained on all data due to the vast amount of features and potential colinearites in the data. Instead, we compared to the relative risk (RR) with a 95% confidence interval (CI). Table 4 shows the five highest and lowest weighted features of zero-shot TabLLM and their relative risk for **EoL**. All top five features have a significantly increased relative risk demonstrating the capabilities of TabLLM to identify relevant features even without any training examples. For the five lowest weighted features, only 'sex_female' has a significantly decreased risk. A list of 100 features is given in Table 17 in the Supplement.

6 DISCUSSION

For all datasets except **Credit-g** and **Heart**, the *List Template* and *Text Template* serializations showed nontrivial zero-shot performance, indicating that TabLLM is able to effectively utilize prior knowledge in the LLM for classification. Serializations with LLMs proved suboptimal due to their noisy outputs suggesting that simple templates are preferable for TabLLM. The performance drops observed when we removed or permuted the column names indicate that the LLM actually makes use of feature names and their relationships to the correct values, especially in the few-shot setting. These findings are partly consistent with Dinh et al. (2022) who used GPT-3 and tested serializations with removed or permuted column names. When using all training examples, they showed that using the correct column names led to the best performance on four classification tasks. In contrast to our results, however, they could not confirm these findings when using only a fraction (0.2, 0.4, 0.6, 0.8) of the training data. A reason for this could be that we tested much fewer number of training examples. In addition to that, we found a very strong drop in performance for permuted values showing that the LLM relies more on the correct values than feature names. Surprisingly, how-

ever, all serializations with less information came close to the best serialization for 256 (tabular datasets) to 1024 training examples (insurance dataset). Hence, when hundreds of training examples are available, the input format proved less relevant, and the LLM was able to adapt (Jin et al., 2022). Like our results, Bertsimas et al. (2022) found that natural language representation of healthcare data gave little-to-no improvement (in their different setup) compared to a more straightforward serialization in the medium-shot setting. Our findings also support prior work showing that irrelevant and even misleading inputs can lead to similar few-shot performance (Min et al., 2022; Webson and Pavlick, 2022; Reynolds and McDonell, 2021). For instance, permuting the column names only showed a difference for up to 16 training examples (see Figure 2).

We found clear performance improvements for TabLLM when using additional training examples. It often outperformed strong baseline models in the very-few-shot setting. This emphasizes the value of leveraging LLMs when only little labeled data is available. Surprisingly, Dinh et al. (2022) could not confirm these findings for GPT-3. On two binary classification tasks a fine-tuned GPT-3 model performed worse than LR for up to 250 training examples. Our results indicate that the sample efficiency of TabLLM is highly task-dependent. The performance on **Blood**, **Credit-g**, **Diabetes**, and **Heart** is worse than the performance on **Income** and **Car**. Most features of the latter datasets have semantically meaningful textual values likely boosting TabLLM's performance. However, TabLLM also achieved reasonable results on numerical datasets (**Blood**, **California**, **Diabetes**, and **Jungle**). In addition, **Diabetes** and **Heart** have somewhat specialized feature names and values, such as "ventricular hypertrophy" and "Plasma glucose concentration," whereas **Income** and **Car** are more general-domain knowledge. This indicates that T0, the language model we used in TabLLM, seems to have less prior knowledge about medicine than about general-domain concepts. Indeed, the training tasks for T0 do not contain any tasks with medical data (Sanh et al., 2022).

Our findings on the three insurance claims datasets partly reinforce this hypothesis. Zero-shot performance depends on the concept selection strategy and the LLM seems to have little knowledge about medical procedures. Prior work has shown that medical-domain-specific language models, such as PubMedBERT, and general-domain models with medical data in their training sets, such as GPT-3, perform well at downstream prediction tasks on medical data even with fairly few samples (Gu et al., 2021; Agrawal et al., 2022). Substituting T0 with one of these models in TabLLM to study medical predictions tasks is an interesting direction for future work.

Our results on the public **Blood**, **Diabetes**, and **Heart** datasets are very similar to our results for **EoL**, **Surgery**, and **LoH**, which are practically relevant but rely on private data. Except for the zero-shot and very few-shot regime, other baselines tend to outperform TabLLM on these datasets. This suggests that **Blood**, **Diabetes**, and **Heart** datasets could be good proxies for the community to further study medical-domain tabular classification with LLMs without needing access to large private datasets.

7 LIMITATIONS AND CONCLUSION

TabLLM has a much larger computational footprint compared to traditional algorithms. It still requires fairly large GPUs to fine-tune the LLM, and inference with T0 requires far more FLOPs than inference with XGBoost or LR. Our results indicate that TabLLM trades off this computational efficiency for improved sample efficiency. Further, as we saw with the three healthcare claims tasks, performance may suffer if the dense feature set for a given row cannot fit within the token limit for a given LLM. Since the gains from TabLLM stem from its ability to use existing domain knowledge, the semantics of the column names and feature values need to have been observed during the LLM's original pre-training. For example, if the columns represent genes, we may not expect a vanilla LLM to have strong representations for gene names. Finally, due to dataset shift, the pre-training data for a given LLM may not necessarily reflect the settings under which a given table was aggregated, e.g., due to inflation and a changing value of money (see Sec. 5 in the Supplement).

Despite these limitations, our empirical results show that TabLLM enjoys strong performance at tabular classification, outperforming state-of-the-art baseline algorithms like XGBoost and SAINT by over 5 AUC points in the very-few-shot regime, all while staying competitive with these methods when a large number of samples is available.

Currently, TabLLM does not use any *unlabeled* data; a fruitful direction could involve leveraging unlabeled data, e.g., using the techniques from Lang et al. (2022) to combine the few-shot performance of TabLLM with the ultimate performance of tree-based baselines by co-training the models together. Other improvements could include more faithful LLM serializations as well as numeric-specific encoding methods (Gorishniy et al., 2022).

8 SOCIETAL IMPACT

Similar to other ML systems that were trained on historic data, LLMs are prone to replicate existing biases and stereotypes. Hence, when applying TabLLM for sensitive tasks such as income or a health trajectory, predictions should be considered with great care and further analyses (e.g., for subgroups) are mandatory. In addition, LLMs require a lot of computing resources. This bears the risk of creating an exclusive research environment. Also, the environmental impact of LLMs can be significant.

9 ACKNOWLEDGEMENTS

SH was supported by the German Academic Exchange Service, HL by NSF AiTF award CCF-1723344, MA by a Takeda Fellowship, and DS, HL, AB, and SH in part by Independence Blue Cross. Thanks to Dr. Steven Horng for generously donating GPU-time on the BIDMC computing cluster (Horng, 2022) and to NVIDIA Corporation for their donation of two NVIDIA A100 GPUs used in this work.

References

Agrawal, M., Hegselmann, S., Lang, H., Kim, Y., and Sontag, D. (2022). Large Language Models are Zero-Shot Clinical Information Extractors. Technical Report arXiv:2205.12689, arXiv.

Arik, S. Ö. and Pfister, T. (2021). Tabnet: Attentive interpretable tabular learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):6679–6687.

Avati, A., Jung, K., Harman, S., Downing, L., Ng, A., and Shah, N. H. (2018). Improving palliative care with deep learning. *BMC medical informatics and decision making*, 18(4):55–64.

Bach, S., Sanh, V., Yong, Z. X., Webson, A., Raffel, C., Nayak, N. V., Sharma, A., Kim, T., Bari, M. S., Fevry, T., Alyafeai, Z., Dey, M., Santilli, A., Sun, Z., Bendavid, S., Xu, C., Chhablani, G., Wang, H., Fries, J., Al-shaibani, M., Sharma, S., Thakker, U., Almubarak, K., Tang, X., Radev, D., Jiang, M. T.-j., and Rush, A. (2022). PromptSource: An integrated development environment and repository for natural language prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104, Dublin, Ireland. Association for Computational Linguistics.

Bahri, D., Jiang, H., Tay, Y., and Metzler, D. (2022). Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*.

Bertsimas, D., Carballo, K. V., Ma, Y., Na, L., Boussioux, L., Zeng, C., Soenksen, L. R., and Fuentes, I. (2022). Tabtext: a systematic approach to aggregate knowledge across tabular data structures. *arXiv preprint arXiv:2206.10381*.

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2022a). Deep Neural Networks and Tabular Data: A Survey. Technical Report arXiv:2110.01889, arXiv.

Borisov, V., Seßler, K., Leemann, T., Pawelczyk, M., and Kasneci, G. (2022b). Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. Association for Computing Machinery.

Cui, G., Hu, S., Ding, N., Huang, L., and Liu, Z. (2022). Prototypical verbalizer for prompt-based few-shot tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7014–7024, Dublin, Ireland. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dinh, T., Zeng, Y., Zhang, R., Lin, Z., Gira, M., Rajput, S., yong Sohn, J., Papailiopoulos, D., and Lee, K. (2022). LIFT: Language-interfaced fine-tuning for non-language machine learning tasks. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*.

Gorishniy, Y., Rubachev, I., and Babenko, A. (2022). On embeddings for numerical features in tabular deep learning. *arXiv preprint arXiv:2203.05556*.

Grinsztajn, L., Oyallon, E., and Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., and Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Haendel, M., Vasilevsky, N., Unni, D., Bologa, C., Harris, N., Rehm, H., Hamosh, A., Baynam, G., Groza, T., McMurry, J., et al. (2020). How many rare diseases are there? *Nature Reviews Drug Discovery*, 19(2):77–78.

Harari, A. and Katz, G. (2022). Few-shot tabular data enrichment using fine-tuned transformer architectures. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1577–1591.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. (2022). Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*.

Horng, S. (2022). Machine Learning Core.

Huang, X., Khetan, A., Cvitkovic, M., and Karnin, Z. (2020). TabTransformer: Tabular Data Modeling Using Contextual Embeddings. Technical Report arXiv:2012.06678, arXiv.

Jin, W., Cheng, Y., Shen, Y., Chen, W., and Ren, X. (2022). A good prompt is worth millions of parameters? low-resource prompt-based learning for vision-language models. In *ACL 2022*.

Kadra, A., Lindauer, M., Hutter, F., and Grabocka, J. (2021). Well-tuned simple nets excel on tabular datasets. *Advances in neural information processing systems*, 34:23928–23941.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Kodialam, R., Boiarsky, R., Lim, J., Sai, A., Dixit, N., and Sontag, D. (2021). Deep contextual clinical prediction with reverse distillation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):249–258.

Kontschieder, P., Fiterau, M., Criminisi, A., and Bulo, S. R. (2015). Deep Neural Decision Forests. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1467–1475, Santiago, Chile. IEEE.

Lang, H., Agrawal, M. N., Kim, Y., and Sontag, D. (2022). Co-training improves prompt-based learning for large language models. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11985–12003. PMLR.

Levin, R., Cherepanova, V., Schwarzschild, A., Bansal, A., Bruss, C. B., Goldstein, T., Wilson, A. G., and Goldblum, M. (2022). Transfer Learning with Deep Tabular Models. Technical Report arXiv:2206.15306, arXiv.

Li, Y., Li, J., Suhara, Y., Doan, A., and Tan, W.-C. (2020). Deep entity matching with pre-trained language models. *Proc. VLDB Endow.*, 14(1):50–60.

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. (2022). Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning. *arXiv:2205.05638 [cs]*. arXiv: 2205.05638.

Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., and Tang, J. (2021). GPT Understands, Too. Technical Report arXiv:2103.10385, arXiv.

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Narayan, A., Chami, I., Orr, L., and Ré, C. (2022). Can Foundation Models Wrangle Your Data? Technical Report arXiv:2205.09911, arXiv.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. *arXiv:2203.02155 [cs]*. arXiv: 2203.02155.

Perez, E., Kiela, D., and Cho, K. (2021). True few-shot learning with language models. *Advances in Neural Information Processing Systems*, 34:11054–11070.

Popov, S., Morozov, S., and Babenko, A. (2020). Neural oblivious decision ensembles for deep learning on tabular data. In *International Conference on Learning Representations*.

Reynolds, L. and McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7.

Sahakyan, M., Aung, Z., and Rahwan, T. (2021). Explainable artificial intelligence for tabular data: A survey. *IEEE Access*, 9:135392–135422.

Sanh, V., Webson, A., Raffel, C., Bach, S., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Raja, A., Dey, M., Bari, M. S., Xu, C., Thakker, U., Sharma, S. S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N., Datta, D., Chang, J., Jiang, M. T.-J., Wang, H., Manica, M., Shen, S., Yong, Z. X., Pandey, H., Bawden, R., Wang, T., Neeraj, T., Rozen, J., Sharma, A., Santilli, A., Fevry, T., Fries, J. A., Teehan, R., Scao, T. L., Biderman, S., Gao, L., Wolf, T., and Rush, A. M. (2022). Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Schick, T. and Schütze, H. (2021). Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Shwartz-Ziv, R. and Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, 81.

Somepalli, G., Goldblum, M., Schwarzschild, A., Bruss, C. B., and Goldstein, T. (2021). SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training. Technical Report arXiv:2106.01342, arXiv.

Webson, A. and Pavlick, E. (2022). Do Prompt-Based Models Really Understand the Meaning of Their Prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344, Seattle, United States. Association for Computational Linguistics.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. (2022). Chain of Thought Prompting Elicits Reasoning in Large Language Models. *arXiv:2201.11903 [cs]*. arXiv: 2201.11903.

Yin, P., Neubig, G., Yih, W.-t., and Riedel, S. (2020). TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Yoon, J., Zhang, Y., Jordon, J., and van der Schaar, M. (2020). VIME: Extending the Success of Self- and Semi-supervised Learning to Tabular Domain. In *Advances in Neural Information Processing Systems*, volume 33, pages 11033–11043. Curran Associates, Inc.

Zhao, Z., Wallace, E., Feng, S., Klein, D., and Singh, S. (2021). Calibrate Before Use: Improving Few-shot Performance of Language Models. In *Proceedings of the 38th International Conference on Machine Learning*, pages 12697–12706. PMLR. ISSN: 2640-3498.

Supplementary Materials: TabLLM: Few-shot Classification of Tabular Data with Large Language Models

1 ADDITIONAL DATASET DETAILS

1.1 Public Tabular Datasets

We systematically identified datasets for classification from Kadra et al. (2021), Grinsztajn et al. (2022), Borisov et al. (2022a), and from Kaggle. Each dataset was separated into 80/20 train-test splits. The k labeled examples $\mathcal{D}_k$ were sampled in a class-balanced manner from the training set. We performed experiments for different numbers of trainings examples (shots) ranging from 0 to 512 and the entire dataset (all). To characterize the sensitivity of models to the choice of k labeled examples, we repeated the dataset splitting and sampling procedures for five different seeds and report the mean AUC and standard deviation (SD) across seeds. No hyperparameter tuning was conducted for TabLLM; for baselines, internal cross validation was conducted to choose optimal hyperparameters, and the model was then retrained on all data. We analyzed the following datasets:

- **Bank** (Kadra et al., 2021) contains information of a direct marketing campaign from a Portugese banking institution (Moro et al., 2014). The goal is to predict whether a customer subscribed to a term deposit or not. It consists of 45,211 rows and 16 features; 5,289 labels are positive.
- **Blood** (Kadra et al., 2021) consists of data of a blood transfusion service from Taiwan (Yeh et al., 2009). It contains 4 attributes of 748 donors and the label is representing whether they returned for another donation (178 positive).
- **California** (Grinsztajn et al., 2022) contains eight attributes of 20,640 districts in California and the goal is to predict the median house value in each district (Pace and Barry, 1997). Analogously to Grinsztajn et al. (2022), we created a balanced classification task by predicting whether the house value is below or above the median (10,317 positive).
- **Car** (Kadra et al., 2021) has entries for different cars that are characterized by six attributes; the task is a multiclass classification problem evaluating the state of each car. The dataset contains 1,728 rows, and the four classes have a distribution of 1210, 384, 65, and 69 examples.
- **Credit-g** (Kadra et al., 2021) describes 1,000 people from Germany that want to receive a credit using 20 attributes. The label is to predict whether they have good or bad risk; 700 are classified as good.
- **Diabetes** (from Kaggle[2]) was collected by the National Institute of Diabetes and Digestive and Kidney Diseases (Smith et al., 1988) and contains 768 rows, each corresponding to women of Pima Indian heritage with eight clinical variables. The task is binary classification of whether a person has diabetes; 268 cases are positive.
- **Heart** (from Kaggle[3]) contains data of four different hospitals (Detrano et al., 1989). Each row contains 11 clinical variables of a patient. The task is binary classification of coronary artery disease. Of the 918 patients, 508 are positive.
- **Income** (Kadra et al., 2021; Borisov et al., 2022a) also called Adult contains rows for 48,842 individuals with twelve attributes collected in the 1994 U.S. Census (Kohavi et al., 1996; Dua and Graff, 2017). The task is to predict whether each person has an annual income over $50,000. The dataset has 11,687 positive labels.
- **Jungle** (Kadra et al., 2021) is a collection of 44,819 end game positions of Jungle Chess (van Rijn and Vis, 2014). Each game is described with 6 attributes and the goal is to predict whether the white player will win (23,062 positive).

[2]https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database (06/28/2022)
[3]https://www.kaggle.com/fedesoriano/heart-failure-prediction(06/28/2022)

Table 5: Evaluation of different concept selection methods for the healthcare claims dataset in the zero-shot setting. The last two rows show the performance when concepts where selected based on the lasso path of logistic regression weights, which violates the zero-shot assumption (*).

Method	EoL	Surgery	LoH
Age, sex, and race	0.59	0.57	0.65
Least frequent conditions	0.57	0.64	0.67
Least frequent procedures	0.59	0.59	0.65
Least frequent concepts (cond. + proc.)	0.55	0.55	0.66
Most frequent conditions	0.67	0.66	0.69
Most frequent procedures	0.59	0.58	0.65
Most frequent concepts (cond. + proc.)	0.62	0.61	0.65
Oldest conditions	0.65	0.66	0.69
Oldest procedures	0.59	0.58	0.65
Oldest concepts (cond. + proc.)	0.60	0.60	0.67
Most recent conditions	0.65	0.66	0.69
Most recent procedures	0.55	0.59	0.65
Most recent concepts (cond. + proc.)	0.59	0.60	0.66
Most relevant concepts based on 256 shots*	0.60	0.58	0.69
Most relevant concepts based on 4096 shots*	0.65	0.57	0.68

1.2 Large Healthcare Claims Dataset

The de-identified health claims data set was provided by a large U.S. health insurer. The data is stored in the Observational Medical Outcomes Partnership (OMOP) Common Data Model version 6.0 (Hripcsak et al., 2015). It contains an entry for every encounter a patient has with the health system. Each entry is associated with a date, a visit type (5 total), a medical specialty (216 total), present conditions (14,095 total), and performed procedures (21,184 total). We additionally used the static concepts age, sex, and race at time of prediction.

We studied three different tasks on this dataset with distinct cohorts. For all tasks, we used a six month outcome period and a gap of three months between time of prediction and the outcome window to prevent data leakage. We required patients to have at least one medical visit and to have been actively enrolled in an insurance plan for at least 95% of the last year and the six month outcome window. We used 10% of the data as a holdout set and sampled the k balanced shots with replacement from the remaining data. We chose larger shot sizes, as the tasks are more complex. We only ran the experiments for a single seed due to runtime limitations. We considered the following tasks:

- **End of Life (EoL)**: We predicted the mortality of all patients older than 70 years. This is often used as a surrogate task. For instance, it can improve initiation of palliative care (Avati et al., 2018) and can help to inform close relatives to reduce family distress (Curtis et al., 2016). The final cohort contained 94,972 individuals; 2,424 were positive.
- **Surgical Procedure (Surgery)**: We predicted the need for any surgical procedure. The task is important in determining health care needs and estimating costs. The cohort included 620,382 people of which 243,349 were positive.
- **Likelihood of Hospitalization (LoH)**: We also predicted the likelihood of being hospitalized. Again, this information can help identify needs and estimate costs. The cohort included 612,656 individuals; 22,427 were positive.

1.2.1 More Details on the Serialization

Each serialization begins with the patient's age, sex, and race. For each concept entry that we included, we also added information of the associated visit. This included its date, the type of doctor the patient saw (e.g., dermatology), if an outpatient visit or length of hospitalization if an inpatient visit, and the primary complaint of the associated visit. If a visit was already added to the serialization, we just added the concept to the existing visit entry. For the *List Template* and *Text Template* serializations approximately 40 medical concepts could be added until the token limit of T0 was reached. To explore the effect of fewer information in the input, we also tested the *List Short* serializations were we added only 10 medical concepts to the serialization. Hence, not the entire token limit of the LLM was used. Examples of the *List Template*, *Text Template* and *List Permuted Names* serializations illustrating this structure are given in Sec. 9.1 at the end of the Supplement.

Table 6: Five examples of different concept names for conditions. The first column shows the original name in the healthcare claims dataset using SNOMED codes. A dash illustrates that no mapping was available.

Original name	ICD	MEDCIN	CHV	Simplify (GPT-3)	Jargon (GPT-3)
Seasonal allergic rhinitis	Allergic rhinitis due to pollen	hay fever	hay fever	Allergies	Seasonal allergic rhinitis
Disturbance in speech	Unspecified speech disturbances	speech difficulties	speech impairment	Speech problems	Dysarthria
Congenital duplication of cervix	—	—	double cervix	Double cervix	Congenital duplication of the cervix
Hypertensive retinopathy	Hypertensive retinopathy	hypertensive retinopathy	hypertensive retinopathy	High blood pressure affecting the retina	Retinopathy h-tensa
Malignant neoplasm of liver	Malignant neoplasm of liver, unspecified	malignant neoplasm of liver	liver cancer	Liver cancer	Hepato-ca

Table 7: Evaluation of alternative condition concepts names. International Classification of Diseases (ICD), MEDCIN and the Consumer Health Vocabulary (CHV) are alternative medical terminologies. We also tested shortening, simplifying, and rewriting concepts as medical jargon via GPT-3. None of the alternative concept names showed consistent performance improvement.

Method	EoL	Surgery	LoH
Original concept names (SNOMED)	0.67	0.66	0.69
Map to ICD concept names	0.67	0.67	0.68
Map to MEDCIN concept names	0.67	0.66	0.69
Map to CHV concept names	0.66	0.66	0.69
Shorten longs concepts with GPT-3	0.67	0.66	0.69
Simplify concepts with GPT-3	0.67	0.66	0.70
Medical jargon with GPT-3	0.68	0.67	0.70

1.2.2 Concept Selection

For the healthcare claims dataset, the number of recorded medical concepts per patients usually exceeded T0's token limit. Hence, we had to determine which concepts of a patient should be included during the serialization. We evaluated four different concept selection strategies in the zero-shot setting for the *List Template* serialization. Choosing the least frequent, most frequent, oldest, or most recent concepts per patient. We tested these for all concepts (conditions and procedures), only conditions, or only procedures. For each patient, we ranked all concepts according to one of the above methods and added concepts until the token limit of the LLM was reached. For least frequent and most frequent, we used the earliest visits associated with the selected medical concepts. We used a simple serialization that only contained the patient's age, sex, and race as a baseline for our experiments. We also tested concept selection based on the lasso path of a logistic regression model determined on 256 and 4,096 shots. This violates the few-shot assumption, but we considered it an interesting comparison with the other strategies that select concepts per patient.

The results are given in Table 5. Using the most frequent conditions per patient consistently outperformed all other selection strategies. Frequent conditions might be useful since they reveal the most relevant condition of a patient. Also, they are usually more common allowing more prior knowledge of the LLM. Across all strategies conditions were usually more useful than procedures. This suggests more prior knowledge of conditions. Interestingly, selecting the most frequent conditions is even better than using the concept weights of a LR model trained on 256 or 4,096 shots.

1.2.3 Alternative Concept Names

The healthcare claims dataset used SNOMED concept names for conditions and SNOMED, Healthcare Common Procedure Coding System (HCPCS), International Classification of Diseases (ICD), and Current Procedural Terminology (CPT) concept names for procedures. We tested different concept names to assess their effect on the performance. We used a zero-

Table 8: Hyperparameters for LR model.

Parameter	Values
penalty	'l1', 'l2'
C	100, 10, 1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5

Table 9: Hyperparameters for LightGBM model.

Parameter	Values
num_leaves	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096
lambda_l1	1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1., 10.
lambda_l2	1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1., 10.
learning_rate	0.01, 0.03, 0.1, 0.3

Table 10: Hyperparameters for XGBoost model.

Parameter	Values
max_depth	2, 4, 6, 8, 10, 12
lambda_l1	1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1.
lambda_l2	1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1.
eta	0.01, 0.03, 0.1, 0.3

shot setting with the *List Template* serialization and the most frequent conditions per patient as the best selection strategy determined as described above. Since the selection method only considered conditions, we only used different condition names. We considered three alternative vocabularies in the Unified Medical Language System (UMLS) that covered at least 20% of the condition concepts and offered different names. ICD is a very common medical terminology offering alternative names for conditions. MEDCIN and the Consumer Health Vocabulary (CHV) offer concept names specifically targeted at clinicians or consumers. We mapped the concept via their UMLS identifier. For ICD we were able to map 7,372, for MEDCIN 9,370 and for CHV 3,700 of the 14,095 condition concepts. Alternatively, we explored concept names generated by GPT-3 (Brown et al., 2020). To do so, we used the publicly accessible GPT-3 API (engine *text-davinci-002*) (Ouyang et al., 2022). We considered shortened names for concepts with more than sixty character ("Rewrite this medical condition with at most six words."), simplified concept names ("Write this medical condition in a short form in lay language.") and medical jargon ("Write this medical condition in medical jargon."). For the simplified names and the medical jargon, we provided GPT-3 with a single example for in-context learning. Examples for all alternative concept names except the shortening are given in Table 6.

The results of this experiment are given in Table 7. We used the most frequent concept as a concept selection methods. Based on the best concept selection, we performed additional experiments for alternative concept names. We found no consistent performance difference even though there were considerable differences in the concept names (see Table 6). Surprisingly, TabLLM performs better for **EoL** and **Surgery** using medical jargon to encode concepts.

2 RUNTIME ESTIMATES FOR TABLLM

The TabLLM training time on the **Income** dataset for 64 training examples and 30 epochs with a batch size of 8 was less than 3 minutes. The average inference time for the test set of 10,000 examples with a batch size of 16 was 2 minutes, around 12 ms per example. The training and inference times for the other public datasets were comparable. Due to the larger size of the healthcare claims dataset, it took nearly 4 minutes to train for 64 examples and 10 epochs for **EoL** and was similar for the other two tasks. Inference took approximately 14 minutes for 10,000 examples with a batch size of 16, i.e. around 84 ms per example. The training times scaled linearly in the shot size.

3 PARAMETER TUNING FOR BASELINES

We used the scikit-learn framework to perform cross-validation and parameter tuning for the LR and the tree-based models (Pedregosa et al., 2011). For LR we tried common parameters for the penalty term and regularization strength (see Table 8). We used the same LR parameters for the public tabular datasets and the healthcare claims dataset. For the tree-based models we adopted the hyperparameter ranges from Borisov et al. (2022a) and Grinsztajn et al. (2022). We discretized the

parameter ranges and performed a complete grid search (see Tables 9 and 10).

For the neural baselines SAINT, TabNet, and NODE, we used the setup and suggested hyperparameter ranges in Borisov et al. (2022a). We modified the open-source implementation of these methods[4] to support ingestion of the nine public tabular datasets. We used the hyperparameter-tuning framework Optuna[5] and selected parameters that maximize AUC-ROC across folds. Note that for the 4-shot setting of the **Car** dataset, AUC may not be defined if the selected validation set includes only one label; in this case we used accuracy as our validation metric but report AUC-ROC on the holdout test set. Each neural baseline model was run for 20 trials with Optuna and trained for 100 epochs per hyperparameter settings.

4 COMPARING BASELINE RESULTS TO THE LITERATURE

To assess whether our baseline results match results reported in the literature, we report studies that used the same models.

Bank Dataset. Kadra et al. (2021) trained a XGBoost, TabNet, and NODE baseline on this dataset and achieved a balanced accuracy of 72.7, 70.6, and 74.6. Our experiments for a set of 512 balanced training examples (512 shots) show a better performance for XGBoost than NODE.

Blood Dataset. The XGBoost, TabNet, and NODE baselines trained in Kadra et al. (2021) achieved a balanced accuracy of 62.3, 64.3, 50. Our results for a set of 512 balanced training examples (512 shots) also show a better performance for TabNet than XGBoost. However, in our experiments NODE performs better than XGBoost and not worse.

California Dataset. Borisov et al. (2022a) trained a Linear Model, XGBoost, LightGBM, TabNet, NODE, and SAINT baseline on a regression version of the dataset. They achieved a mean squared error of 0.53, 0.21, 0.20, 0.35, 0.28, and 0.23. Our experiments for a set of 512 balanced training examples (512 shots) show a better performance for XGBoost than LightGBM and the same performance for TabNet and NODE. Also, our linear model performs much better which is probably due to more extensive hyperparameter tuning.

Car Dataset. The XGBoost, TabNet, and NODE models in Kadra et al. (2021) showed a balanced accuracy of 92.4, 98.7, and 46.1. In our experiments, XGBoost and TabNet performed very similar for many training examples and NODE was only slightly inferior.

Credit-g Dataset. The XGBoost, TabNet, and NODE baselines trained in Kadra et al. (2021) achieved a balanced accuracy of 68.9, 61.2, and 73.1. Our AUC results cannot easily be compared but our experiments for 512 balanced training examples (512 shots) follow the same trend.

Diabetes Dataset. Hasan et al. (2020) reported an AUC of 0.828 (0.030) for XGBoost on the diabetes dataset, which matches our findings. With additional feature selection and preprocessing methods they reached an AUC of 0.946 (0.020) with XGBoost, but this was out of the scope of our work. XGBoost was the most performant model that they included in their experiments.

Heart Dataset. Muhammad et al. (2020) used only the 303 instances from the Cleveland cohort, while we combined all four sub-cohorts. They achieved an AUC of 0.923 with LR, which is close to our results on all sub-cohorts. They also tested several models that outperformed LR.

Income Dataset. Many studies used the Income or Adult dataset. The review Borisov et al. (2022a) included several of our baselines. They reported an AUC of 0.854 (0.002) for a linear model, 0.928 (0.001) for XGBoost, 0.928 (0.001) for LightGBM, 0.916 (0.002) for SAINT, 0.911 (0.001) for TabNet, and 0.911 (0.002) for NODE. These are in accordance with our results. We reckon the better performance of our LR model is due to more extensive parameter tuning.

Jungle Dataset. The XGBoost and TabNet baselines trained in Kadra et al. (2021) achieved a balanced accuracy of 87.3 and 73.4. They did not train a NODE moel for this dataset. The results follows the same trend as our experiments for a set of 512 balanced training examples (512 shots).

[4]https://github.com/kathrinse/TabSurvey

[5]https://github.com/optuna/optuna

Table 11: The mean performance for one prompt (ours, SD over five seed omitted) and the mean performance and SD across five different prompts (each again over five seeds).

Dataset	Bank	Blood	California	Car	Credit-g	Diabetes	Heart	Income	Jungle
TabLLM 0-shot: 1 prompt (ours)	0.63	0.61	0.61	0.81	0.53	0.68	0.54	0.84	0.60
TabLLM 0-shot: avg. 5 prompts	$0.64_{.01}$	$0.60_{.02}$	$0.59_{.01}$	$0.80_{.01}$	$0.52_{.01}$	$0.67_{.01}$	$0.55_{.04}$	$0.84_{.01}$	$0.60_{.00}$

5 ADJUSTING INCOME DATASET FOR INFLATION

We wanted to investigate how a distribution shift caused by inflation affects the zero-shot performance of TabLLM. The **Income** dataset was collected in 1994, and the label and two features (capital gain/loss in last year) contain dollar values. T0 was trained in 2021 (Sanh et al., 2022), and we assumed that the training data is much more recent than the **Income** dataset. The inflation rate from 1994 to 2021 is 1.79[6]. Without inflation correction the zero-shot results were 0.80 (0.01). Correcting the two features, correcting only the prompt, and correcting both all yielded the same performance as the uncorrected one. The accuracy values also remained the same with the inflation correction.

6 FEATURE IMPORTANCE ANALYSIS OF TABLLM

We wanted to understand which features were most important for the zero-shot performance of TabLLM on **Income** and **EoL**. To this end, we used zero-shot TabLLM with the *List Template* serialization to predict the label probability of all examples in the dataset. We then used 4-fold cross validation to fit a L2-regularized LR model to the predicted label using the features in the serialization as covariates. For **EoL**, we used age, sex, race, and the conditions as inputs, which summed up to 14,105 features.

For **Income** we compared these approximated importance scores to the feature coefficients of a LR model trained on all data for a single seed (Table 16). We used the same setup for the LR model as for our main experiments. We did 4-fold cross validation on an 80% training split to choose hyperparameters, and then refit the model using all training data. The best parameters of the LR model for **Income** were a 'l1' penalty and a regularization constant of 1. For **EoL**, we decided that the LR model coefficients did not provide a good estimate of the ground truth due to the vast amount of features and possible collinearities in the data. Instead, we provide the relative risk (RR) with 95% confidence intervals (CI) treating the occurrence of a feature as an intervention. We report the 50 most and least important features of TabLLM in Table 17.

7 EFFECT OF USING DIFFERENT PROMPTS

To evaluate the effect of using a different prompt we considered the zero-shot setting, since even few training examples mostly cancel the effect. For all datasets we constructed five different prompts that contained the same question, e.g., "Does this person earn a lot of money?" instead of "Does this person earn more than 50000 dollars per year?" for the **Income** dataset. The results are summarized in Table 11. The effects were relative small ranging from a standard deviation of 0.00 for **Jungle** to 0.04 for **Heart** across the five prompts. This suggests that TabLLM is not very sensitive to using different prompts.

[6]U.S. Bureau of Labor Statistics, CPI Inflation Calculator: https://www.bls.gov/data/inflation_calculator.htm

Table 12: Test AUC performance of competing methods on public tabular datasets. Each column reports the k-shot performance for different values of k. Standard deviations across five random seeds are shown as subscripts.

	Number of Shots									
Method	**0**	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**all**
Bank Dataset										
Logistic regression	—	$0.55_{.09}$	$0.66_{.09}$	$0.75_{.06}$	$0.81_{.02}$	$0.84_{.02}$	$0.86_{.02}$	$0.88_{.01}$	$0.89_{.00}$	$0.91_{.00}$
Logistic regression (ordinal)	—	$0.51_{.02}$	$0.60_{.12}$	$0.68_{.09}$	$0.78_{.04}$	$0.82_{.01}$	$0.84_{.03}$	$0.86_{.01}$	$0.87_{.00}$	$0.88_{.00}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.77_{.03}$	$0.84_{.03}$	$0.88_{.01}$	$0.89_{.00}$	$0.94_{.00}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.78_{.03}$	$0.84_{.02}$	$0.87_{.01}$	$0.89_{.00}$	$0.94_{.00}$
XGBoost	—	$0.50_{.00}$	$0.56_{.09}$	$0.68_{.04}$	$0.76_{.03}$	$0.83_{.02}$	$0.85_{.03}$	$0.88_{.01}$	$0.90_{.01}$	$0.94_{.00}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.56_{.09}$	$0.69_{.05}$	$0.75_{.04}$	$0.82_{.02}$	$0.84_{.03}$	$0.87_{.01}$	$0.89_{.00}$	$0.93_{.00}$
SAINT	—	$0.51_{.10}$	$0.61_{.11}$	$0.70_{.04}$	$0.77_{.03}$	$0.81_{.03}$	$0.85_{.02}$	$0.88_{.01}$	$0.88_{.01}$	$0.93_{.00}$
TabNet	—	$0.51_{.06}$	$0.58_{.05}$	$0.64_{.10}$	$0.62_{.04}$	$0.71_{.06}$	$0.73_{.03}$	$0.80_{.04}$	$0.83_{.03}$	$0.93_{.00}$
NODE	—	$0.52_{.02}$	$0.55_{.06}$	$0.64_{.06}$	$0.73_{.06}$	$0.78_{.02}$	$0.83_{.03}$	$0.85_{.01}$	$0.86_{.01}$	$0.76_{.02}$
TabPFN	—	$0.59_{.14}$	$0.66_{.08}$	$0.69_{.02}$	$0.76_{.03}$	$0.82_{.03}$	$0.86_{.02}$	$0.89_{.00}$	$0.90_{.00}$	$0.91_{.00}$
TabPFN (ordinal)	—	$0.57_{.10}$	$0.67_{.05}$	$0.71_{.05}$	$0.78_{.04}$	$0.83_{.01}$	$0.86_{.02}$	$0.87_{.00}$	$0.88_{.00}$	$0.89_{.00}$
TabLLM (T0 + Text GPT-3)	$0.63_{.01}$	$0.61_{.04}$	$0.62_{.02}$	$0.63_{.03}$	$0.64_{.02}$	$0.66_{.04}$	$0.76_{.04}$	$0.81_{.02}$	$0.82_{.01}$	*
TabLLM (T0 + Text T0)	$0.54_{.01}$	$0.56_{.08}$	$0.60_{.06}$	$0.59_{.06}$	$0.60_{.04}$	$0.62_{.04}$	$0.67_{.04}$	$0.79_{.03}$	$0.85_{.01}$	*
TabLLM (T0 + Table-To-Text)	$0.42_{.01}$	$0.48_{.07}$	$0.50_{.05}$	$0.56_{.03}$	$0.57_{.04}$	$0.59_{.05}$	$0.63_{.03}$	$0.68_{.02}$	$0.74_{.01}$	*
TabLLM (T0 + Text Template)	$0.63_{.01}$	$0.59_{.10}$	$0.64_{.05}$	$0.65_{.05}$	$0.64_{.06}$	$0.69_{.03}$	$0.82_{.05}$	$0.87_{.01}$	$0.88_{.01}$	0.92 †
TabLLM (T0 + List Template)	$0.60_{.01}$	$0.59_{.10}$	$0.66_{.02}$	$0.65_{.04}$	$0.66_{.05}$	$0.74_{.07}$	$0.85_{.02}$	$0.87_{.01}$	$0.87_{.01}$	*
TabLLM (T0 + List Only Values)	$0.56_{.01}$	$0.58_{.09}$	$0.60_{.04}$	$0.63_{.03}$	$0.67_{.03}$	$0.71_{.05}$	$0.79_{.03}$	$0.84_{.01}$	$0.86_{.01}$	*
TabLLM (T0 + List Perm. Names)	$0.64_{.00}$	$0.55_{.10}$	$0.62_{.07}$	$0.63_{.04}$	$0.63_{.05}$	$0.68_{.04}$	$0.82_{.02}$	$0.86_{.01}$	$0.88_{.00}$	*
TabLLM (T0 + List Perm. Values)	$0.38_{.01}$	$0.47_{.11}$	$0.53_{.06}$	$0.55_{.07}$	$0.57_{.05}$	$0.65_{.04}$	$0.75_{.07}$	$0.84_{.01}$	$0.85_{.01}$	*
TabLLM (T0 3B + Text Template)	$0.61_{.01}$	$0.60_{.10}$	$0.65_{.05}$	$0.64_{.07}$	$0.65_{.05}$	$0.70_{.02}$	$0.77_{.05}$	$0.88_{.01}$	$0.89_{.01}$	*
Blood Dataset										
Logistic regression	—	$0.54_{.09}$	$0.59_{.08}$	$0.72_{.03}$	$0.70_{.06}$	$0.74_{.02}$	$0.76_{.02}$	$0.76_{.02}$	$0.76_{.03}$	$0.76_{.03}$
Logistic regression (ordinal)	—	$0.54_{.09}$	$0.59_{.08}$	$0.72_{.03}$	$0.70_{.06}$	$0.74_{.02}$	$0.76_{.02}$	$0.76_{.02}$	$0.76_{.03}$	$0.76_{.03}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.69_{.04}$	$0.71_{.05}$	$0.71_{.07}$	$0.67_{.05}$	$0.74_{.04}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.69_{.04}$	$0.71_{.05}$	$0.71_{.07}$	$0.67_{.05}$	$0.74_{.04}$
XGBoost	—	$0.50_{.00}$	$0.58_{.07}$	$0.66_{.04}$	$0.67_{.06}$	$0.68_{.05}$	$0.71_{.06}$	$0.70_{.07}$	$0.67_{.06}$	$0.71_{.04}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.58_{.07}$	$0.66_{.04}$	$0.67_{.06}$	$0.68_{.05}$	$0.71_{.06}$	$0.70_{.07}$	$0.67_{.06}$	$0.71_{.04}$
SAINT	—	$0.47_{.12}$	$0.66_{.08}$	$0.66_{.03}$	$0.67_{.06}$	$0.67_{.05}$	$0.71_{.03}$	$0.76_{.05}$	$0.73_{.02}$	$0.74_{.03}$
TabNet	—	$0.47_{.09}$	$0.61_{.06}$	$0.60_{.09}$	$0.66_{.06}$	$0.63_{.06}$	$0.66_{.04}$	$0.72_{.06}$	$0.72_{.02}$	$0.71_{.03}$
NODE	—	$0.49_{.04}$	$0.60_{.07}$	$0.62_{.04}$	$0.67_{.03}$	$0.71_{.05}$	$0.76_{.03}$	$0.74_{.03}$	$0.76_{.03}$	$0.74_{.03}$
TabPFN	—	$0.52_{.08}$	$0.64_{.04}$	$0.67_{.01}$	$0.70_{.04}$	$0.73_{.04}$	$0.75_{.04}$	$0.76_{.04}$	$0.76_{.03}$	$0.74_{.03}$
TabPFN (ordinal)	—	$0.52_{.08}$	$0.64_{.04}$	$0.67_{.01}$	$0.70_{.04}$	$0.73_{.04}$	$0.75_{.04}$	$0.76_{.04}$	$0.76_{.03}$	$0.74_{.03}$
TabLLM (T0 + Text GPT-3)	$0.63_{.04}$	$0.61_{.07}$	$0.65_{.04}$	$0.63_{.02}$	$0.64_{.03}$	$0.62_{.05}$	$0.67_{.06}$	$0.68_{.05}$	$0.66_{.05}$	*
TabLLM (T0 + Text T0)	$0.49_{.04}$	$0.51_{.03}$	$0.59_{.08}$	$0.59_{.06}$	$0.64_{.04}$	$0.65_{.06}$	$0.66_{.05}$	$0.68_{.06}$	$0.66_{.03}$	*
TabLLM (T0 + Table-To-Text)	$0.61_{.04}$	$0.59_{.04}$	$0.59_{.03}$	$0.57_{.03}$	$0.62_{.07}$	$0.56_{.07}$	$0.57_{.07}$	$0.64_{.07}$	$0.61_{.05}$	*
TabLLM (T0 + Text Template)	$0.61_{.04}$	$0.58_{.09}$	$0.66_{.03}$	$0.66_{.07}$	$0.68_{.04}$	$0.68_{.04}$	$0.68_{.06}$	$0.70_{.08}$	$0.68_{.04}$	$0.70_{.04}$
TabLLM (T0 + List Template)	$0.56_{.05}$	$0.54_{.08}$	$0.64_{.02}$	$0.64_{.08}$	$0.67_{.05}$	$0.66_{.06}$	$0.67_{.05}$	$0.70_{.06}$	$0.67_{.06}$	*
TabLLM (T0 + List Only Values)	$0.45_{.05}$	$0.49_{.07}$	$0.57_{.03}$	$0.57_{.06}$	$0.62_{.06}$	$0.61_{.04}$	$0.64_{.04}$	$0.68_{.07}$	$0.67_{.05}$	*
TabLLM (T0 + List Perm. Names)	$0.52_{.04}$	$0.49_{.07}$	$0.62_{.03}$	$0.62_{.06}$	$0.65_{.05}$	$0.65_{.04}$	$0.68_{.06}$	$0.72_{.06}$	$0.68_{.04}$	*
TabLLM (T0 + List Perm. Values)	$0.51_{.03}$	$0.51_{.06}$	$0.54_{.04}$	$0.52_{.07}$	$0.55_{.03}$	$0.59_{.06}$	$0.59_{.02}$	$0.62_{.06}$	$0.62_{.05}$	*
TabLLM (T0 3B + Text Template)	$0.42_{.05}$	$0.47_{.04}$	$0.62_{.04}$	$0.62_{.09}$	$0.65_{.07}$	$0.67_{.04}$	$0.69_{.04}$	$0.71_{.06}$	$0.67_{.04}$	*
California Dataset										
Logistic regression	—	$0.58_{.11}$	$0.69_{.13}$	$0.80_{.06}$	$0.84_{.03}$	$0.88_{.01}$	$0.90_{.00}$	$0.91_{.00}$	$0.91_{.00}$	$0.92_{.00}$
Logistic regression (ordinal)	—	$0.58_{.11}$	$0.69_{.13}$	$0.80_{.06}$	$0.84_{.03}$	$0.88_{.01}$	$0.90_{.00}$	$0.91_{.00}$	$0.91_{.00}$	$0.92_{.00}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.81_{.02}$	$0.87_{.01}$	$0.90_{.01}$	$0.92_{.00}$	$0.97_{.00}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.81_{.02}$	$0.87_{.01}$	$0.90_{.01}$	$0.92_{.00}$	$0.97_{.00}$
XGBoost	—	$0.50_{.00}$	$0.62_{.10}$	$0.74_{.03}$	$0.79_{.04}$	$0.82_{.04}$	$0.87_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.97_{.00}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.62_{.10}$	$0.74_{.03}$	$0.79_{.04}$	$0.82_{.04}$	$0.87_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.97_{.00}$
SAINT	—	$0.59_{.09}$	$0.64_{.12}$	$0.73_{.06}$	$0.76_{.06}$	$0.81_{.02}$	$0.84_{.01}$	$0.88_{.02}$	$0.91_{.02}$	$0.95_{.00}$
TabNet	—	$0.50_{.08}$	$0.57_{.06}$	$0.67_{.02}$	$0.69_{.05}$	$0.72_{.03}$	$0.79_{.02}$	$0.84_{.02}$	$0.87_{.01}$	$0.96_{.00}$
NODE	—	$0.58_{.06}$	$0.57_{.07}$	$0.70_{.05}$	$0.77_{.03}$	$0.80_{.01}$	$0.86_{.02}$	$0.86_{.02}$	$0.87_{.01}$	$0.87_{.01}$
TabPFN	—	$0.63_{.13}$	$0.63_{.11}$	$0.80_{.03}$	$0.85_{.03}$	$0.89_{.01}$	$0.91_{.01}$	$0.92_{.00}$	$0.93_{.00}$	$0.94_{.00}$
TabPFN (ordinal)	—	$0.63_{.13}$	$0.63_{.11}$	$0.80_{.03}$	$0.85_{.03}$	$0.89_{.01}$	$0.91_{.01}$	$0.92_{.00}$	$0.93_{.00}$	$0.94_{.00}$
TabLLM (T0 + Text GPT-3)	$0.56_{.00}$	$0.55_{.03}$	$0.57_{.05}$	$0.61_{.06}$	$0.73_{.05}$	$0.73_{.04}$	$0.82_{.01}$	$0.84_{.01}$	$0.85_{.01}$	*
TabLLM (T0 + Text T0)	$0.49_{.01}$	$0.52_{.02}$	$0.51_{.02}$	$0.52_{.02}$	$0.54_{.04}$	$0.56_{.04}$	$0.69_{.02}$	$0.73_{.03}$	$0.80_{.02}$	*
TabLLM (T0 + Table-To-Text)	$0.49_{.01}$	$0.50_{.01}$	$0.51_{.01}$	$0.52_{.02}$	$0.57_{.04}$	$0.58_{.04}$	$0.74_{.03}$	$0.79_{.02}$	$0.82_{.01}$	*
TabLLM (T0 + Text Template)	$0.61_{.01}$	$0.63_{.05}$	$0.60_{.07}$	$0.70_{.08}$	$0.77_{.08}$	$0.77_{.04}$	$0.81_{.02}$	$0.83_{.01}$	$0.86_{.02}$	$0.95_{.00}$
TabLLM (T0 + List Template)	$0.61_{.01}$	$0.64_{.05}$	$0.62_{.06}$	$0.68_{.07}$	$0.77_{.07}$	$0.79_{.02}$	$0.82_{.02}$	$0.84_{.01}$	$0.87_{.01}$	*
TabLLM (T0 + List Only Values)	$0.58_{.01}$	$0.57_{.08}$	$0.55_{.03}$	$0.65_{.09}$	$0.74_{.08}$	$0.77_{.03}$	$0.83_{.01}$	$0.84_{.02}$	$0.86_{.02}$	*
TabLLM (T0 + List Perm. Names)	$0.54_{.01}$	$0.52_{.03}$	$0.52_{.04}$	$0.52_{.03}$	$0.66_{.06}$	$0.74_{.01}$	$0.81_{.02}$	$0.84_{.02}$	$0.86_{.02}$	*
TabLLM (T0 + List Perm. Values)	$0.47_{.01}$	$0.48_{.02}$	$0.50_{.01}$	$0.52_{.02}$	$0.57_{.03}$	$0.64_{.04}$	$0.71_{.04}$	$0.76_{.01}$	$0.78_{.02}$	*
TabLLM (T0 3B + Text Template)	$0.57_{.01}$	$0.59_{.03}$	$0.57_{.04}$	$0.66_{.07}$	$0.77_{.06}$	$0.79_{.02}$	$0.81_{.01}$	$0.83_{.01}$	$0.85_{.01}$	*

* Result omitted due to runtime limitations of TabLLM on the full dataset.
† Only a single run performed due to runtime limitations of TabLLM on the full dataset.

Table 13: Test AUC performance of competing methods on public tabular datasets. Each column reports the k-shot performance for different values of k. Standard deviations across five random seeds are shown as subscripts.

	Number of Shots									
Method	**0**	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**all**
Car Dataset										
Logistic regression	—	$0.61_{.02}$	$0.65_{.10}$	$0.74_{.07}$	$0.83_{.02}$	$0.93_{.02}$	$0.96_{.01}$	$0.97_{.01}$	$0.98_{.00}$	$0.98_{.00}$
Logistic regression (ordinal)	—	$0.62_{.06}$	$0.63_{.05}$	$0.64_{.07}$	$0.75_{.04}$	$0.73_{.03}$	$0.73_{.03}$	$0.74_{.03}$	$0.76_{.02}$	$0.78_{.03}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.85_{.06}$	$0.93_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.75_{.04}$	$0.91_{.05}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$
XGBoost	—	$0.50_{.00}$	$0.59_{.04}$	$0.70_{.08}$	$0.82_{.03}$	$0.91_{.02}$	$0.95_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.55_{.03}$	$0.70_{.04}$	$0.78_{.03}$	$0.90_{.03}$	$0.94_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
SAINT	—	$0.56_{.08}$	$0.64_{.08}$	$0.76_{.03}$	$0.85_{.03}$	$0.92_{.02}$	$0.96_{.01}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$
TabNet	—	†	$0.54_{.05}$	$0.64_{.05}$	$0.66_{.05}$	$0.73_{.07}$	$0.81_{.04}$	$0.93_{.02}$	$0.98_{.01}$	$1.00_{.00}$
NODE	—	$0.51_{.10}$	$0.57_{.06}$	$0.69_{.02}$	$0.74_{.03}$	$0.80_{.02}$	$0.82_{.01}$	$0.91_{.01}$	$0.96_{.01}$	$0.93_{.01}$
TabPFN	—	$0.64_{.06}$	$0.75_{.05}$	$0.87_{.04}$	$0.92_{.02}$	$0.97_{.00}$	$0.99_{.01}$	$1.00_{.00}$	$1.00_{.00}$	$1.00_{.00}$
TabPFN (ordinal)	—	$0.59_{.06}$	$0.65_{.08}$	$0.75_{.04}$	$0.82_{.06}$	$0.89_{.01}$	$0.93_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
TabLLM (T0 + Text GPT-3)	$0.72_{.02}$	$0.75_{.03}$	$0.75_{.02}$	$0.78_{.01}$	$0.83_{.01}$	$0.87_{.02}$	$0.90_{.01}$	$0.93_{.02}$	$0.93_{.02}$	$0.96_{.01}$
TabLLM (T0 + Text T0)	$0.85_{.01}$	$0.85_{.02}$	$0.84_{.03}$	$0.86_{.02}$	$0.89_{.02}$	$0.92_{.02}$	$0.94_{.01}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$
TabLLM (T0 + Table-To-Text)	$0.61_{.01}$	$0.69_{.04}$	$0.74_{.04}$	$0.79_{.02}$	$0.88_{.01}$	$0.91_{.02}$	$0.94_{.01}$	$0.96_{.01}$	$0.95_{.01}$	$0.96_{.00}$
TabLLM (T0 + Text Template)	$0.82_{.02}$	$0.83_{.03}$	$0.85_{.03}$	$0.86_{.03}$	$0.91_{.02}$	$0.96_{.02}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$	$1.00_{.00}$
TabLLM (T0 + List Template)	$0.79_{.02}$	$0.84_{.03}$	$0.85_{.02}$	$0.86_{.03}$	$0.91_{.02}$	$0.95_{.01}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$	$1.00_{.00}$
TabLLM (T0 + List Only Values)	$0.48_{.03}$	$0.62_{.04}$	$0.67_{.03}$	$0.70_{.03}$	$0.75_{.02}$	$0.87_{.02}$	$0.94_{.01}$	$0.98_{.01}$	$0.99_{.01}$	$1.00_{.00}$
TabLLM (T0 + List Perm. Names)	$0.39_{.02}$	$0.54_{.10}$	$0.58_{.06}$	$0.70_{.03}$	$0.86_{.02}$	$0.94_{.01}$	$0.97_{.02}$	$0.99_{.01}$	$0.99_{.00}$	$1.00_{.00}$
TabLLM (T0 + List Perm. Values)	$0.38_{.02}$	$0.48_{.08}$	$0.55_{.05}$	$0.63_{.04}$	$0.69_{.03}$	$0.78_{.02}$	$0.90_{.03}$	$0.98_{.01}$	$1.00_{.00}$	$1.00_{.00}$
TabLLM (T0 3B + Text Template)	$0.78_{.02}$	$0.80_{.03}$	$0.84_{.03}$	$0.84_{.04}$	$0.89_{.03}$	$0.91_{.01}$	$0.96_{.01}$	$0.98_{.01}$	$0.99_{.00}$	$1.00_{.00}$
Credit-g Dataset										
Logistic regression	—	$0.50_{.08}$	$0.56_{.06}$	$0.58_{.08}$	$0.68_{.08}$	$0.66_{.07}$	$0.71_{.06}$	$0.75_{.04}$	$0.76_{.02}$	$0.79_{.03}$
Logistic regression (ordinal)	—	$0.56_{.05}$	$0.54_{.06}$	$0.55_{.05}$	$0.61_{.05}$	$0.68_{.05}$	$0.66_{.03}$	$0.68_{.04}$	$0.71_{.02}$	$0.72_{.02}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.61_{.09}$	$0.68_{.03}$	$0.72_{.02}$	$0.75_{.02}$	$0.78_{.02}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.68_{.07}$	$0.66_{.04}$	$0.72_{.02}$	$0.75_{.03}$	$0.76_{.04}$
XGBoost	—	$0.50_{.00}$	$0.51_{.07}$	$0.59_{.05}$	$0.66_{.03}$	$0.67_{.06}$	$0.68_{.02}$	$0.73_{.02}$	$0.75_{.03}$	$0.78_{.04}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.54_{.11}$	$0.57_{.08}$	$0.64_{.05}$	$0.66_{.06}$	$0.68_{.04}$	$0.74_{.02}$	$0.76_{.03}$	$0.76_{.04}$
SAINT	—	$0.56_{.08}$	$0.53_{.05}$	$0.60_{.05}$	$0.66_{.06}$	$0.66_{.06}$	$0.68_{.05}$	$0.72_{.04}$	$0.73_{.03}$	$0.77_{.04}$
TabNet	—	$0.48_{.05}$	$0.52_{.07}$	$0.49_{.03}$	$0.52_{.03}$	$0.56_{.05}$	$0.60_{.05}$	$0.61_{.02}$	$0.66_{.04}$	$0.64_{.03}$
NODE	—	$0.54_{.09}$	$0.54_{.10}$	$0.54_{.09}$	$0.59_{.07}$	$0.63_{.04}$	$0.68_{.02}$	$0.68_{.05}$	$0.70_{.02}$	$0.65_{.03}$
TabPFN	—	$0.58_{.08}$	$0.59_{.03}$	$0.64_{.06}$	$0.69_{.07}$	$0.70_{.07}$	$0.72_{.06}$	$0.75_{.04}$	$0.75_{.02}$	$0.75_{.03}$
TabPFN (ordinal)	—	$0.55_{.08}$	$0.51_{.07}$	$0.57_{.06}$	$0.62_{.03}$	$0.66_{.05}$	$0.70_{.02}$	$0.73_{.01}$	$0.73_{.03}$	$0.75_{.04}$
TabLLM (T0 + Text GPT-3)	$0.52_{.04}$	$0.53_{.04}$	$0.56_{.03}$	$0.56_{.05}$	$0.55_{.05}$	$0.57_{.08}$	$0.60_{.06}$	$0.61_{.04}$	$0.63_{.05}$	*
TabLLM (T0 + Text T0)	$0.49_{.02}$	$0.50_{.06}$	$0.54_{.06}$	$0.55_{.04}$	$0.60_{.06}$	$0.61_{.02}$	$0.61_{.02}$	$0.63_{.03}$	$0.65_{.02}$	*
TabLLM (T0 + Table-To-Text)	$0.50_{.06}$	$0.65_{.04}$	$0.60_{.05}$	$0.60_{.07}$	$0.65_{.05}$	$0.67_{.05}$	$0.65_{.05}$	$0.68_{.04}$	$0.64_{.05}$	*
TabLLM (T0 + Text Template)	$0.53_{.05}$	$0.69_{.04}$	$0.66_{.04}$	$0.66_{.05}$	$0.72_{.06}$	$0.70_{.07}$	$0.71_{.07}$	$0.72_{.03}$	$0.72_{.02}$	$0.70_{.02}$
TabLLM (T0 + List Template)	$0.53_{.05}$	$0.64_{.04}$	$0.60_{.06}$	$0.64_{.05}$	$0.70_{.05}$	$0.66_{.08}$	$0.67_{.03}$	$0.70_{.03}$	$0.70_{.04}$	*
TabLLM (T0 + List Only Values)	$0.66_{.06}$	$0.71_{.03}$	$0.67_{.06}$	$0.69_{.06}$	$0.72_{.06}$	$0.69_{.05}$	$0.69_{.07}$	$0.70_{.06}$	$0.68_{.04}$	*
TabLLM (T0 + List Perm. Names)	$0.44_{.01}$	$0.58_{.09}$	$0.59_{.08}$	$0.60_{.07}$	$0.70_{.06}$	$0.69_{.06}$	$0.67_{.05}$	$0.70_{.05}$	$0.70_{.03}$	*
TabLLM (T0 + List Perm. Values)	$0.50_{.05}$	$0.55_{.06}$	$0.56_{.07}$	$0.58_{.04}$	$0.64_{.03}$	$0.66_{.08}$	$0.67_{.09}$	$0.68_{.03}$	$0.69_{.03}$	*
TabLLM (T0 3B + Text Template)	$0.54_{.03}$	$0.65_{.05}$	$0.63_{.05}$	$0.63_{.03}$	$0.73_{.04}$	$0.69_{.05}$	$0.68_{.06}$	$0.73_{.05}$	$0.73_{.03}$	*
Diabetes Dataset										
Logistic regression	—	$0.60_{.15}$	$0.68_{.11}$	$0.73_{.05}$	$0.76_{.05}$	$0.80_{.02}$	$0.81_{.02}$	$0.83_{.02}$	$0.83_{.02}$	$0.83_{.02}$
Logistic regression (ordinal)	—	$0.60_{.15}$	$0.68_{.11}$	$0.73_{.05}$	$0.76_{.05}$	$0.80_{.02}$	$0.81_{.02}$	$0.83_{.02}$	$0.83_{.02}$	$0.83_{.02}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.79_{.02}$	$0.79_{.04}$	$0.79_{.02}$	$0.79_{.03}$	$0.83_{.03}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.79_{.02}$	$0.79_{.04}$	$0.79_{.02}$	$0.79_{.03}$	$0.83_{.03}$
XGBoost	—	$0.50_{.00}$	$0.59_{.16}$	$0.72_{.07}$	$0.69_{.08}$	$0.73_{.05}$	$0.78_{.05}$	$0.80_{.03}$	$0.80_{.01}$	$0.84_{.03}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.59_{.16}$	$0.72_{.07}$	$0.69_{.08}$	$0.73_{.05}$	$0.78_{.05}$	$0.80_{.03}$	$0.80_{.01}$	$0.84_{.03}$
SAINT	—	$0.46_{.12}$	$0.65_{.11}$	$0.73_{.06}$	$0.73_{.06}$	$0.79_{.03}$	$0.81_{.03}$	$0.81_{.04}$	$0.77_{.03}$	$0.83_{.03}$
TabNet	—	$0.56_{.04}$	$0.56_{.06}$	$0.64_{.09}$	$0.66_{.06}$	$0.71_{.04}$	$0.73_{.04}$	$0.74_{.05}$	$0.74_{.07}$	$0.81_{.03}$
NODE	—	$0.49_{.13}$	$0.67_{.09}$	$0.69_{.08}$	$0.73_{.05}$	$0.77_{.04}$	$0.80_{.04}$	$0.81_{.03}$	$0.83_{.02}$	$0.83_{.03}$
TabPFN	—	$0.61_{.13}$	$0.67_{.11}$	$0.71_{.07}$	$0.77_{.03}$	$0.82_{.03}$	$0.83_{.03}$	$0.83_{.03}$	$0.81_{.02}$	$0.81_{.03}$
TabPFN (ordinal)	—	$0.61_{.13}$	$0.67_{.11}$	$0.71_{.07}$	$0.77_{.03}$	$0.82_{.03}$	$0.83_{.03}$	$0.83_{.03}$	$0.81_{.02}$	$0.81_{.03}$
TabLLM (T0 + Text GPT-3)	$0.61_{.06}$	$0.61_{.07}$	$0.56_{.12}$	$0.67_{.08}$	$0.74_{.04}$	$0.77_{.02}$	$0.79_{.03}$	$0.76_{.03}$	$0.78_{.04}$	$0.81_{.04}$
TabLLM (T0 + Text T0)	$0.58_{.04}$	$0.53_{.05}$	$0.53_{.06}$	$0.54_{.09}$	$0.59_{.05}$	$0.68_{.02}$	$0.73_{.04}$	$0.72_{.05}$	$0.72_{.03}$	$0.76_{.01}$
TabLLM (T0 + Table-To-Text)	$0.58_{.04}$	$0.51_{.10}$	$0.53_{.07}$	$0.56_{.05}$	$0.57_{.04}$	$0.59_{.04}$	$0.72_{.05}$	$0.74_{.04}$	$0.75_{.06}$	$0.77_{.04}$
TabLLM (T0 + Text Template)	$0.68_{.06}$	$0.61_{.09}$	$0.63_{.08}$	$0.69_{.07}$	$0.68_{.04}$	$0.73_{.03}$	$0.79_{.04}$	$0.78_{.02}$	$0.78_{.04}$	$0.80_{.04}$
TabLLM (T0 + List Template)	$0.64_{.06}$	$0.64_{.09}$	$0.64_{.10}$	$0.67_{.07}$	$0.70_{.05}$	$0.76_{.04}$	$0.78_{.03}$	$0.78_{.03}$	$0.78_{.04}$	$0.81_{.05}$
TabLLM (T0 + List Only Values)	$0.55_{.05}$	$0.54_{.07}$	$0.52_{.05}$	$0.59_{.08}$	$0.63_{.04}$	$0.67_{.07}$	$0.73_{.03}$	$0.75_{.06}$	$0.77_{.04}$	$0.79_{.03}$
TabLLM (T0 + List Perm. Names)	$0.56_{.07}$	$0.60_{.09}$	$0.68_{.12}$	$0.74_{.05}$	$0.74_{.03}$	$0.72_{.04}$	$0.76_{.04}$	$0.77_{.04}$	$0.77_{.04}$	$0.81_{.04}$
TabLLM (T0 + List Perm. Values)	$0.44_{.03}$	$0.47_{.09}$	$0.43_{.06}$	$0.55_{.07}$	$0.61_{.05}$	$0.65_{.05}$	$0.73_{.03}$	$0.76_{.03}$	$0.78_{.02}$	$0.80_{.03}$
TabLLM (T0 3B + Text Template)	$0.62_{.05}$	$0.57_{.07}$	$0.60_{.08}$	$0.67_{.05}$	$0.67_{.06}$	$0.76_{.03}$	$0.77_{.04}$	$0.81_{.05}$	$0.80_{.04}$	$0.82_{.04}$

* Result omitted due to runtime limitations of TabLLM on the full dataset.
† Result omitted due to TabNet package not supporting unseen labels in validation set during cross validation.

Table 14: Test AUC performance of competing methods on public tabular datasets. Each column reports the k-shot performance for different values of k. Standard deviations across five random seeds are shown as subscripts.

	Number of Shots									
Method	**0**	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**all**
Heart Dataset										
Logistic regression	—	$0.69_{.17}$	$0.75_{.13}$	$0.82_{.06}$	$0.87_{.05}$	$0.91_{.01}$	$0.90_{.02}$	$0.92_{.01}$	$0.93_{.01}$	$0.93_{.01}$
Logistic regression (ordinal)	—	$0.70_{.17}$	$0.73_{.14}$	$0.84_{.04}$	$0.88_{.03}$	$0.89_{.01}$	$0.88_{.02}$	$0.90_{.02}$	$0.92_{.02}$	$0.92_{.02}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.91_{.01}$	$0.91_{.01}$	$0.91_{.01}$	$0.93_{.00}$	$0.94_{.01}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.91_{.01}$	$0.91_{.02}$	$0.91_{.01}$	$0.92_{.01}$	$0.94_{.01}$
XGBoost	—	$0.50_{.00}$	$0.55_{.14}$	$0.84_{.07}$	$0.88_{.04}$	$0.91_{.01}$	$0.91_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.94_{.01}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.56_{.15}$	$0.84_{.07}$	$0.90_{.03}$	$0.91_{.01}$	$0.90_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.94_{.01}$
SAINT	—	$0.80_{.12}$	$0.83_{.10}$	$0.88_{.07}$	$0.90_{.01}$	$0.90_{.04}$	$0.90_{.02}$	$0.90_{.01}$	$0.92_{.01}$	$0.93_{.01}$
TabNet	—	$0.56_{.12}$	$0.70_{.05}$	$0.73_{.14}$	$0.80_{.04}$	$0.83_{.05}$	$0.84_{.03}$	$0.88_{.02}$	$0.88_{.03}$	$0.89_{.03}$
NODE	—	$0.52_{.10}$	$0.78_{.08}$	$0.83_{.03}$	$0.86_{.02}$	$0.88_{.02}$	$0.88_{.01}$	$0.91_{.02}$	$0.92_{.03}$	$0.92_{.03}$
TabPFN	—	$0.84_{.06}$	$0.88_{.05}$	$0.87_{.06}$	$0.91_{.02}$	$0.92_{.02}$	$0.92_{.02}$	$0.92_{.01}$	$0.92_{.02}$	$0.92_{.02}$
TabPFN (ordinal)	—	$0.79_{.08}$	$0.85_{.07}$	$0.88_{.05}$	$0.90_{.02}$	$0.92_{.01}$	$0.92_{.01}$	$0.92_{.00}$	$0.92_{.02}$	$0.92_{.02}$
TabLLM (T0 + Text GPT-3)	$0.51_{.04}$	$0.72_{.05}$	$0.82_{.03}$	$0.85_{.05}$	$0.88_{.03}$	$0.91_{.02}$	$0.89_{.02}$	$0.91_{.01}$	$0.91_{.01}$	$0.93_{.01}$
TabLLM (T0 + Text T0)	$0.44_{.03}$	$0.74_{.07}$	$0.82_{.10}$	$0.87_{.02}$	$0.88_{.02}$	$0.89_{.04}$	$0.90_{.01}$	$0.89_{.02}$	$0.89_{.03}$	$0.93_{.02}$
TabLLM (T0 + Table-To-Text)	$0.56_{.05}$	$0.73_{.09}$	$0.78_{.08}$	$0.86_{.06}$	$0.88_{.03}$	$0.91_{.02}$	$0.91_{.02}$	$0.90_{.02}$	$0.91_{.01}$	$0.92_{.01}$
TabLLM (T0 + Text Template)	$0.54_{.04}$	$0.76_{.14}$	$0.83_{.05}$	$0.87_{.04}$	$0.87_{.06}$	$0.91_{.01}$	$0.90_{.01}$	$0.92_{.01}$	$0.92_{.01}$	$0.94_{.01}$
TabLLM (T0 + List Template)	$0.52_{.03}$	$0.73_{.12}$	$0.83_{.05}$	$0.87_{.04}$	$0.88_{.04}$	$0.91_{.02}$	$0.91_{.01}$	$0.92_{.01}$	$0.92_{.01}$	$0.94_{.01}$
TabLLM (T0 + List Only Values)	$0.40_{.04}$	$0.67_{.16}$	$0.83_{.06}$	$0.84_{.05}$	$0.88_{.03}$	$0.89_{.03}$	$0.92_{.02}$	$0.90_{.00}$	$0.90_{.01}$	$0.92_{.01}$
TabLLM (T0 + List Perm. Names)	$0.57_{.02}$	$0.78_{.07}$	$0.85_{.02}$	$0.82_{.06}$	$0.87_{.05}$	$0.90_{.02}$	$0.92_{.02}$	$0.91_{.01}$	$0.91_{.01}$	$0.93_{.02}$
TabLLM (T0 + List Perm. Values)	$0.23_{.02}$	$0.63_{.20}$	$0.79_{.12}$	$0.83_{.07}$	$0.88_{.04}$	$0.89_{.04}$	$0.90_{.02}$	$0.91_{.01}$	$0.91_{.01}$	$0.93_{.00}$
TabLLM (T0 3B + Text Template)	$0.56_{.03}$	$0.68_{.13}$	$0.82_{.04}$	$0.85_{.02}$	$0.86_{.03}$	$0.90_{.01}$	$0.91_{.01}$	$0.93_{.01}$	$0.93_{.01}$	$0.94_{.01}$
Income Dataset										
Logistic regression	—	$0.68_{.15}$	$0.72_{.13}$	$0.80_{.03}$	$0.82_{.01}$	$0.83_{.03}$	$0.85_{.01}$	$0.87_{.01}$	$0.88_{.00}$	$0.90_{.00}$
Logistic regression (ordinal)	—	$0.55_{.04}$	$0.56_{.06}$	$0.58_{.07}$	$0.70_{.06}$	$0.76_{.03}$	$0.79_{.01}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.78_{.03}$	$0.81_{.03}$	$0.87_{.01}$	$0.88_{.00}$	$0.93_{.00}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.78_{.01}$	$0.81_{.01}$	$0.86_{.01}$	$0.89_{.00}$	$0.93_{.00}$
XGBoost	—	$0.50_{.00}$	$0.59_{.06}$	$0.77_{.02}$	$0.79_{.03}$	$0.82_{.02}$	$0.84_{.01}$	$0.87_{.01}$	$0.88_{.00}$	$0.93_{.00}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.63_{.04}$	$0.74_{.04}$	$0.76_{.04}$	$0.79_{.03}$	$0.84_{.02}$	$0.86_{.01}$	$0.88_{.00}$	$0.93_{.00}$
SAINT	—	$0.74_{.03}$	$0.65_{.15}$	$0.79_{.03}$	$0.81_{.03}$	$0.84_{.02}$	$0.84_{.02}$	$0.87_{.01}$	$0.88_{.00}$	$0.91_{.00}$
TabNet	—	$0.56_{.04}$	$0.59_{.07}$	$0.62_{.11}$	$0.64_{.06}$	$0.71_{.04}$	$0.73_{.05}$	$0.80_{.02}$	$0.83_{.02}$	$0.92_{.00}$
NODE	—	$0.54_{.02}$	$0.54_{.04}$	$0.65_{.04}$	$0.67_{.03}$	$0.75_{.02}$	$0.78_{.01}$	$0.78_{.01}$	$0.83_{.01}$	$0.82_{.00}$
TabPFN	—	$0.73_{.08}$	$0.71_{.09}$	$0.76_{.09}$	$0.80_{.04}$	$0.82_{.04}$	$0.84_{.01}$	$0.86_{.01}$	$0.87_{.01}$	$0.89_{.00}$
TabPFN (ordinal)	—	$0.64_{.11}$	$0.64_{.06}$	$0.72_{.04}$	$0.77_{.02}$	$0.80_{.02}$	$0.81_{.01}$	$0.83_{.01}$	$0.85_{.01}$	$0.87_{.00}$
TabLLM (T0 + Text GPT-3)	$0.75_{.01}$	$0.79_{.03}$	$0.80_{.03}$	$0.82_{.02}$	$0.82_{.01}$	$0.84_{.02}$	$0.84_{.02}$	$0.85_{.01}$	$0.86_{.00}$	*
TabLLM (T0 + Text T0)	$0.65_{.01}$	$0.67_{.03}$	$0.66_{.07}$	$0.72_{.02}$	$0.75_{.03}$	$0.79_{.04}$	$0.82_{.02}$	$0.83_{.02}$	$0.86_{.01}$	*
TabLLM (T0 + Table-To-Text)	$0.50_{.00}$	$0.64_{.07}$	$0.64_{.11}$	$0.72_{.05}$	$0.74_{.03}$	$0.79_{.03}$	$0.81_{.01}$	$0.84_{.01}$	$0.84_{.01}$	*
TabLLM (T0 + Text Template)	$0.84_{.00}$	$0.84_{.01}$	$0.84_{.02}$	$0.84_{.04}$	$0.84_{.01}$	$0.84_{.02}$	$0.86_{.01}$	$0.87_{.00}$	$0.89_{.01}$	$0.92_{.00}$
TabLLM (T0 + List Template)	$0.79_{.01}$	$0.83_{.01}$	$0.83_{.03}$	$0.83_{.02}$	$0.84_{.01}$	$0.85_{.01}$	$0.86_{.01}$	$0.87_{.01}$	$0.88_{.01}$	*
TabLLM (T0 + List Only Values)	$0.73_{.01}$	$0.74_{.04}$	$0.75_{.04}$	$0.80_{.03}$	$0.82_{.01}$	$0.84_{.01}$	$0.84_{.01}$	$0.86_{.01}$	$0.87_{.01}$	*
TabLLM (T0 + List Perm. Names)	$0.65_{.00}$	$0.75_{.03}$	$0.74_{.05}$	$0.82_{.02}$	$0.83_{.02}$	$0.84_{.02}$	$0.86_{.01}$	$0.86_{.01}$	$0.88_{.01}$	*
TabLLM (T0 + List Perm. Values)	$0.26_{.00}$	$0.40_{.04}$	$0.48_{.10}$	$0.65_{.06}$	$0.72_{.03}$	$0.79_{.03}$	$0.81_{.02}$	$0.83_{.01}$	$0.84_{.01}$	*
TabLLM (T0 3B + Text Template)	$0.76_{.00}$	$0.77_{.06}$	$0.80_{.04}$	$0.83_{.02}$	$0.83_{.03}$	$0.85_{.01}$	$0.86_{.00}$	$0.86_{.01}$	$0.88_{.01}$	*
Jungle Dataset										
Logistic regression	—	$0.62_{.09}$	$0.69_{.09}$	$0.68_{.04}$	$0.76_{.03}$	$0.79_{.01}$	$0.79_{.00}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
Logistic regression (ordinal)	—	$0.62_{.09}$	$0.69_{.09}$	$0.68_{.04}$	$0.76_{.03}$	$0.79_{.01}$	$0.79_{.00}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
LightGBM	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.79_{.02}$	$0.84_{.02}$	$0.88_{.01}$	$0.91_{.00}$	$0.98_{.00}$
LightGBM (ordinal)	—	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.50_{.00}$	$0.79_{.02}$	$0.84_{.02}$	$0.88_{.01}$	$0.91_{.00}$	$0.98_{.00}$
XGBoost	—	$0.50_{.00}$	$0.58_{.07}$	$0.72_{.05}$	$0.78_{.03}$	$0.81_{.02}$	$0.84_{.02}$	$0.87_{.01}$	$0.91_{.01}$	$0.98_{.00}$
XGBoost (ordinal)	—	$0.50_{.00}$	$0.58_{.07}$	$0.72_{.05}$	$0.78_{.03}$	$0.81_{.02}$	$0.84_{.02}$	$0.87_{.01}$	$0.91_{.01}$	$0.98_{.00}$
SAINT	—	$0.64_{.05}$	$0.69_{.06}$	$0.72_{.05}$	$0.79_{.02}$	$0.81_{.01}$	$0.83_{.01}$	$0.88_{.01}$	$0.90_{.00}$	$1.00_{.00}$
TabNet	—	$0.53_{.09}$	$0.60_{.05}$	$0.62_{.03}$	$0.69_{.04}$	$0.73_{.04}$	$0.75_{.02}$	$0.79_{.02}$	$0.84_{.01}$	$0.99_{.00}$
NODE	—	$0.60_{.01}$	$0.71_{.03}$	$0.68_{.04}$	$0.74_{.02}$	$0.75_{.04}$	$0.78_{.01}$	$0.79_{.01}$	$0.80_{.00}$	$0.81_{.00}$
TabPFN	—	$0.65_{.08}$	$0.72_{.04}$	$0.71_{.07}$	$0.78_{.02}$	$0.81_{.01}$	$0.84_{.01}$	$0.88_{.01}$	$0.91_{.00}$	$0.93_{.00}$
TabPFN (ordinal)	—	$0.65_{.08}$	$0.72_{.04}$	$0.71_{.07}$	$0.78_{.02}$	$0.81_{.01}$	$0.84_{.01}$	$0.88_{.01}$	$0.91_{.00}$	$0.93_{.00}$
TabLLM (T0 + Text GPT-3)	$0.56_{.01}$	$0.58_{.02}$	$0.55_{.02}$	$0.60_{.06}$	$0.68_{.03}$	$0.74_{.03}$	$0.77_{.01}$	$0.81_{.01}$	$0.85_{.01}$	*
TabLLM (T0 + Text T0)	$0.63_{.00}$	$0.63_{.04}$	$0.64_{.05}$	$0.62_{.06}$	$0.70_{.01}$	$0.71_{.03}$	$0.74_{.02}$	$0.78_{.02}$	$0.82_{.01}$	*
TabLLM (T0 + Table-To-Text)	$0.51_{.01}$	$0.60_{.02}$	$0.60_{.04}$	$0.63_{.05}$	$0.69_{.03}$	$0.75_{.01}$	$0.78_{.03}$	$0.82_{.01}$	$0.85_{.01}$	*
TabLLM (T0 + Text Template)	$0.60_{.00}$	$0.64_{.01}$	$0.64_{.02}$	$0.65_{.03}$	$0.71_{.02}$	$0.78_{.02}$	$0.81_{.02}$	$0.84_{.01}$	$0.89_{.01}$	1.00 †
TabLLM (T0 + List Template)	$0.63_{.00}$	$0.65_{.01}$	$0.66_{.03}$	$0.66_{.04}$	$0.71_{.03}$	$0.78_{.02}$	$0.81_{.03}$	$0.84_{.01}$	$0.88_{.01}$	*
TabLLM (T0 + List Only Values)	$0.58_{.00}$	$0.60_{.03}$	$0.62_{.03}$	$0.63_{.02}$	$0.65_{.04}$	$0.73_{.01}$	$0.76_{.02}$	$0.82_{.02}$	$0.88_{.01}$	*
TabLLM (T0 + List Perm. Names)	$0.40_{.00}$	$0.53_{.06}$	$0.55_{.05}$	$0.63_{.10}$	$0.72_{.03}$	$0.79_{.02}$	$0.80_{.03}$	$0.84_{.02}$	$0.89_{.01}$	*
TabLLM (T0 + List Perm. Values)	$0.48_{.00}$	$0.50_{.02}$	$0.52_{.03}$	$0.53_{.03}$	$0.55_{.01}$	$0.59_{.02}$	$0.63_{.01}$	$0.72_{.02}$	$0.75_{.01}$	*
TabLLM (T0 3B + Text Template)	$0.54_{.00}$	$0.63_{.02}$	$0.64_{.04}$	$0.67_{.03}$	$0.72_{.03}$	$0.77_{.02}$	$0.80_{.02}$	$0.83_{.01}$	$0.87_{.01}$	*

* Result omitted due to runtime limitations of TabLLM on the full dataset.

† These experiments were only performed for a single run due to runtime limitations of TabLLM on the full dataset.

Table 15: Full results on healthcare claims dataset. The best concept selection method (most frequent concepts) and concept names (original concept names) were used as determined in prior zero-shot experiments. A fix number of 10 epochs was used for up to 256 shots and 3 epochs for more shots to decrease the runtime and prevent overfitting.

	Number of Shots							
Method	**0**	**16**	**64**	**256**	**1,024**	**4,096**	**16,384**	**all**
End of Life (EoL)								
TabLLM (T0 + List Template)	0.70	0.74	0.78	0.78	0.79	0.81	0.81	—
TabLLM (T0 + Text Template)	0.63	0.71	0.74	0.76	0.78	0.79	0.80	—
TabLLM (T0 + List Short)	0.68	0.71	0.76	0.79	0.80	0.81	0.82	—
TabLLM (T0 + List Perm. Names)	0.62	0.66	0.70	0.74	0.75	0.77	0.79	—
Logistic Regression	—	$0.65_{.07}$	$0.77_{.02}$	$0.80_{.02}$	$0.83_{.01}$	$0.83_{.01}$	$0.84_{.01}$	$0.84_{.01}$
LightGBM	—	$0.50_{.00}$	$0.71_{.01}$	$0.76_{.02}$	$0.80_{.01}$	$0.82_{.01}$	$0.83_{.01}$	0.82 *
TabLLM (T0 + List Template) unbalanced	0.70	0.64	0.69	0.74	0.74	0.77	0.79	—
Logistic Regression unbalanced	—	$0.44_{.04}$	$0.53_{.12}$	$0.75_{.03}$	$0.77_{.03}$	$0.80_{.02}$	$0.82_{.02}$	$0.84_{.01}$
Surgical Procedure (Surgery)								
TabLLM (T0 + List Template)	0.67	0.73	0.72	0.73	0.75	0.78	0.79	—
TabLLM (T0 + Text Template)	0.62	0.71	0.69	0.72	0.74	0.77	0.78	—
TabLLM (T0 + List Short)	0.66	0.70	0.69	0.72	0.73	0.76	0.78	—
TabLLM (T0 + List Perm. Names)	0.60	0.68	0.70	0.72	0.74	0.77		—
Logistic Regression	—	$0.72_{.04}$	$0.75_{.05}$	$0.77_{.01}$	$0.79_{.01}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
LightGBM	—	$0.50_{.00}$	$0.73_{.02}$	$0.77_{.01}$	$0.79_{.01}$	$0.80_{.00}$	$0.81_{.01}$	0.82 *
TabLLM (T0 + List Template) unbalanced	0.67	0.68	0.73	0.74	0.75	0.77	0.79	—
Logistic Regression unbalanced	—	$0.61_{.15}$	$0.77_{.01}$	$0.77_{.02}$	$0.78_{.01}$	$0.80_{.01}$	$0.80_{.00}$	$0.81_{.00}$
Likelihood of Hospitalization (LoH)								
TabLLM (T0 + List Template)	0.71	0.73	0.73	0.76	0.78	0.81	0.82	—
TabLLM (T0 + Text Template)	0.65	0.74	0.72	0.74	0.78	0.80	0.81	—
TabLLM (T0 + List Short)	0.70	0.73	0.75	0.78	0.79	0.80	0.82	—
TabLLM (T0 + List Perm. Names)	0.62	0.71	0.72	0.75	0.75	0.78	0.80	—
Logistic Regression	—	$0.72_{.04}$	$0.76_{.03}$	$0.80_{.01}$	$0.82_{.01}$	$0.83_{.01}$	$0.83_{.01}$	$0.84_{.01}$
LightGBM	—	$0.50_{.00}$	$0.72_{.02}$	$0.76_{.03}$	$0.81_{.01}$	$0.83_{.00}$	$0.83_{.01}$	0.85 *
TabLLM (T0 + List Template) unbalanced	0.71	0.66	0.72	0.75	0.75	0.78	0.80	—
Logistic Regression unbalanced	—	$0.53_{.06}$	$0.54_{.09}$	$0.73_{.06}$	$0.79_{.01}$	$0.81_{.01}$	$0.82_{.01}$	$0.84_{.01}$

* These experiments were only performed for a single run due to runtime limitations on the full dataset.

Table 16: Feature importance of zero-shot TabLLM and LR on all data for the **Income** dataset. To determine the feature importance of TabLLM, we fit a separate LR model to the predictions using the original feature values as covariates. For LR we simply use the feature coefficients. The features are ranked by their TabLLM importance score.

Feature	TabLLM		LR		Feature	TabLLM		LR	
	rank	weight	rank	weight		rank	weight	rank	weight
capital_gain	1	5.310	2	2.393	relationship_Other-relative	54	-0.010	88	-0.759
education_Masters	2	4.623	6	1.455	native_country_Trinadad&Tob.	55	-0.028	66	-0.097
education_Doctorate	3	3.410	4	2.066	race_Black	56	-0.044	74	-0.291
education_Bachelors	4	2.995	7	1.135	native_country_England	57	-0.088	16	0.551
education_Prof-school	5	2.949	5	1.900	native_country_Honduras	58	-0.105	58	0.000
occupation_Machine-op-insp.	6	2.589	75	-0.325	relationship_Not-in-family	59	-0.153	29	0.257
workclass_Private	7	2.275	37	0.102	native_country_Holand-Neth.	60	-0.154	57	0.000
relationship_Wife	8	2.109	8	0.955	occupation_Craft-repair	61	-0.161	36	0.108
native_country_China	9	2.086	94	-0.839	capital_loss	62	-0.182	31	0.255
native_country_United-States	10	2.045	38	0.087	race_Other	63	-0.202	65	-0.085
native_country_Taiwan	11	1.965	54	0.000	native_country_Yugoslavia	64	-0.204	27	0.357
workclass_Federal-gov	12	1.784	14	0.574	workclass_Local-gov	65	-0.230	47	0.000
race_White	13	1.685	61	0.000	occupation_nan	66	-0.248	82	-0.653
education_Assoc-acdm	14	1.621	13	0.574	marital_status_Never-married	67	-0.292	77	-0.443
native_country_nan	15	1.565	63	-0.056	native_country_Iran	68	-0.330	41	0.000
marital_status_Married-civ-sp.	16	1.487	3	2.214	native_country_Dominican-Rep.	69	-0.332	85	-0.731
occupation_Protective-serv	17	1.434	17	0.535	marital_status_Married-sp.-abs.	70	-0.379	51	0.000
sex_Male	18	1.335	42	0.000	native_country_Jamaica	71	-0.416	25	0.392
occupation_Armed-Forces	19	1.290	60	0.000	native_country_Nicaragua	72	-0.425	45	0.000
occupation_Adm-clerical	20	1.245	52	0.000	native_country_Thailand	73	-0.451	100	-1.116
hours_per_week	21	1.240	20	0.424	native_country_Peru	74	-0.522	93	-0.837
native_country_Hong	22	1.227	86	-0.749	native_country_Japan	75	-0.617	56	0.000
occupation_Tech-support	23	1.164	18	0.526	relationship_Unmarried	76	-0.620	48	0.000
relationship_Husband	24	1.087	72	-0.212	native_country_France	77	-0.754	21	0.416
occupation_Sales	25	0.857	28	0.298	occupation_Other-service	78	-0.754	96	-0.903
native_country_Vietnam	26	0.803	95	-0.898	workclass_Never-worked	79	-0.763	50	0.000
marital_status_Married-AF-sp.	27	0.792	1	2.571	education_1st-4th	80	-0.763	101	-1.172
native_country_Philippines	28	0.711	40	0.011	native_country_Columbia	81	-0.836	104	-1.855
age	29	0.710	22	0.411	education_5th-6th	82	-0.843	97	-0.961
native_country_Poland	30	0.698	53	0.000	marital_status_Divorced	83	-0.870	46	0.000
occupation_Prof-specialty	31	0.684	12	0.620	education_9th	84	-0.904	102	-1.222
race_Asian-Pac-Islander	32	0.651	32	0.254	native_country_Ecuador	85	-0.952	49	0.000
native_country_Outlying-US	33	0.591	92	-0.836	education_11th	86	-0.993	91	-0.825
workclass_Self-emp-not-inc	34	0.582	76	-0.344	native_country_Haiti	87	-1.062	35	0.137
native_country_Italy	35	0.534	24	0.400	education_Assoc-voc	88	-1.074	19	0.514
marital_status_Separated	36	0.523	70	-0.181	native_country_India	89	-1.074	71	-0.183
workclass_nan	37	0.515	59	0.000	education_7th-8th	90	-1.151	103	-1.303
occupation_Exec-managerial	38	0.503	10	0.773	marital_status_Widowed	91	-1.253	64	-0.071
native_country_Scotland	39	0.491	81	-0.626	education_10th	92	-1.306	89	-0.797
native_country_Laos	40	0.475	44	0.000	native_country_Greece	93	-1.319	68	-0.140
native_country_Cambodia	41	0.328	11	0.642	sex_Female	94	-1.327	84	-0.710
native_country_Guatemala	42	0.276	55	0.000	native_country_South	95	-1.466	99	-1.101
workclass_State-gov	43	0.267	73	-0.223	native_country_Cuba	96	-1.575	33	0.230
native_country_Germany	44	0.262	39	0.043	education_Some-college	97	-1.950	26	0.363
native_country_Puerto-Rico	45	0.241	67	-0.128	occupation_Handlers-cleaners	98	-1.992	83	-0.681
native_country_Hungary	46	0.177	34	0.191	native_country_Portugal	99	-2.049	15	0.572
native_country_Mexico	47	0.123	80	-0.579	race_Amer-Indian-Eskimo	100	-2.081	78	-0.465
native_country_Ireland	48	0.116	9	0.954	relationship_Own-child	101	-2.404	87	-0.755
education_HS-grad	49	0.092	43	0.000	occupation_Priv-house-serv	102	-2.840	105	-1.909
occupation_Transport-moving	50	0.090	62	-0.048	education_12th	103	-3.178	79	-0.480
native_country_El-Salvador	51	0.027	90	-0.803	education_Preschool	104	-3.520	106	-2.385
native_country_Canada	52	0.027	23	0.407	occupation_Farming-fishing	105	-3.853	98	-0.982
workclass_Self-emp-inc	53	0.001	30	0.255	workclass_Without-pay	106	-4.423	69	-0.174

Table 17: Feature importance of zero-shot TabLLM and relative risk (RR) with 95% confidence interval (CI) for **EoL** task on the healthcare claims dataset. For TabLLM we fit a separate LR model to the predictions using the original feature values as covariates. We determine the relative risk treating the respective feature as an intervention, i.e. the ratio of the label in the group that has a concept divided by the ratio in the group without it. We selected 50 features with the highest and the lowest importance.

Feature	TabLLM rank	TabLLM weight	RR (95% CI)	Feature	TabLLM rank	TabLLM weight	RR (95% CI)
atrial fibrillation	1	0.633	2.72 (2.51-2.95)	open wound of forehead without ...	14056	-0.152	1.80 (1.18-2.74)
atherosclerosis of coronary art...	2	0.530	2.10 (1.94-2.27)	prediabetes	14057	-0.157	0.81 (0.68-0.96)
atherosclerosis of aorta	3	0.473	1.99 (1.81-2.19)	primary iridocyclitis	14058	-0.157	1.63 (1.03-2.56)
exudative age-related macular d...	4	0.452	2.38 (2.06-2.75)	discoloration of skin	14059	-0.157	0.87 (0.73-1.04)
sex_male	5	0.442	1.23 (1.14-1.33)	basal cell carcinoma of truncal...	14060	-0.158	1.14 (0.94-1.40)
non-hodgkin's lymphoma (clinical)	6	0.440	1.36 (0.94-1.96)	lumbar sprain	14061	-0.158	1.14 (0.91-1.42)
chronic atrial fibrillation	7	0.436	3.36 (3.05-3.70)	spasm	14062	-0.160	0.98 (0.82-1.16)
chronic kidney disease stage 3	8	0.430	2.75 (2.53-2.98)	chronic rhinitis	14063	-0.161	1.22 (1.06-1.42)
atherosclerosis of arteries of ...	9	0.404	2.76 (2.42-3.15)	primary cardiomyopathy	14064	-0.161	2.50 (2.11-2.97)
barrett's esophagus	10	0.402	1.07 (0.84-1.37)	benign neoplastic disease	14065	-0.162	1.04 (0.63-1.72)
chronic obstructive lung disease	11	0.401	2.39 (2.19-2.60)	palpitations	14066	-0.166	1.12 (1.01-1.25)
paroxysmal atrial fibrillation	12	0.395	2.58 (2.37-2.81)	localized, primary osteoarthrit...	14067	-0.167	1.50 (1.33-1.70)
systemic lupus erythematosus	13	0.395	1.51 (0.99-2.29)	benign neoplasm of skin of lowe...	14068	-0.167	0.68 (0.53-0.89)
atherosclerosis of artery of lo...	14	0.394	2.45 (2.20-2.72)	cyst of ovary	14069	-0.171	0.90 (0.64-1.26)
coronary atherosclerosis	15	0.381	2.15 (1.95-2.36)	microscopic hematuria	14070	-0.171	1.18 (1.01-1.37)
nonexudative age-related macula...	16	0.377	2.15 (1.95-2.37)	problem related to lifestyle	14071	-0.172	0.96 (0.48-1.91)
age related macular degeneration	17	0.371	2.18 (1.76-2.71)	acquired hypothyroidism	14072	-0.172	1.47 (1.34-1.62)
pseudoexfoliation glaucoma	18	0.360	1.13 (0.72-1.76)	abnormal findings on diagnostic...	14073	-0.176	0.63 (0.54-0.73)
degenerative joint disease invo...	19	0.359	1.77 (1.52-2.06)	increased frequency of urination	14074	-0.177	1.41 (1.22-1.64)
coronary arteriosclerosis	20	0.357	2.00 (1.82-2.20)	disorder of skin	14075	-0.178	1.18 (0.95-1.48)
coronary artery graft present	21	0.346	1.64 (1.41-1.91)	thyroiditis	14076	-0.180	0.87 (0.49-1.57)
aortocoronary bypass graft present	22	0.335	2.24 (1.98-2.54)	race_hispanic_or_latino	14077	-0.186	0.96 (0.60-1.51)
dehydration	23	0.332	2.94 (2.68-3.22)	herpes zoster without complication	14078	-0.187	1.14 (0.96-1.35)
primary malignant neoplasm of f...	24	0.327	1.19 (1.01-1.40)	altered sensation of skin	14079	-0.191	1.00 (0.82-1.22)
malignant lymphoma	25	0.322	1.54 (0.96-2.46)	generalized hyperhidrosis	14080	-0.194	1.37 (1.07-1.76)
cerebral infarction due to thro...	26	0.316	2.86 (2.46-3.32)	primary open angle glaucoma	14081	-0.194	1.35 (1.20-1.52)
congestive heart failure	27	0.313	3.67 (3.38-3.99)	stool finding	14082	-0.195	1.48 (1.26-1.73)
old myocardial infarction	28	0.299	2.04 (1.81-2.30)	primary gout	14083	-0.196	1.80 (1.51-2.15)
sleep apnea	29	0.294	1.16 (0.98-1.37)	localized, primary osteoarthrit...	14084	-0.199	1.10 (0.92-1.30)
acute hypoxemic respiratory fai...	30	0.292	4.02 (3.62-4.46)	diarrhea	14085	-0.200	1.73 (1.57-1.90)
obstructive sleep apnea syndrome	31	0.287	1.09 (0.96-1.24)	benign neoplasm of skin of uppe...	14086	-0.204	0.78 (0.58-1.03)
primary malignant neoplasm of e...	32	0.284	0.92 (0.56-1.53)	prostatitis	14087	-0.204	1.20 (0.89-1.62)
sensorineural hearing loss	33	0.281	1.26 (1.09-1.47)	eruption	14088	-0.205	1.25 (1.11-1.41)
retention of urine	34	0.280	2.19 (1.97-2.44)	scar conditions and fibrosis of...	14089	-0.206	1.00 (0.86-1.15)
atrial flutter	35	0.280	2.14 (1.85-2.47)	hashimoto thyroiditis	14090	-0.215	0.91 (0.49-1.68)
abdominal aortic aneurysm witho...	36	0.275	1.85 (1.58-2.18)	acquired deformity of toe	14091	-0.227	1.25 (0.94-1.65)
chronic kidney disease due to h...	37	0.274	2.65 (2.42-2.90)	race_asian	14092	-0.228	0.70 (0.50-0.99)
non-rheumatic aortic sclerosis	38	0.271	2.64 (2.38-2.93)	localized swelling, mass and lu...	14093	-0.242	1.48 (1.15-1.91)
type 2 diabetes mellitus	39	0.267	2.14 (1.96-2.33)	benign neoplasm of skin of trunk	14094	-0.245	0.91 (0.79-1.05)
intraductal carcinoma in situ o...	40	0.265	0.62 (0.30-1.29)	benign essential hypertension	14095	-0.245	1.86 (1.72-2.01)
chronic kidney disease stage 2	41	0.264	1.77 (1.55-2.03)	finding of frequency of urination	14096	-0.255	1.48 (1.34-1.64)
degenerative disorder of macula	42	0.263	2.23 (1.88-2.65)	benign essential microscopic he...	14097	-0.258	1.10 (0.76-1.59)
sensorineural hearing loss, bil...	43	0.262	1.30 (1.17-1.43)	localized swelling, mass and lu...	14098	-0.262	1.93 (1.67-2.23)
race_white	44	0.262	1.25 (1.14-1.37)	digestive symptom	14099	-0.267	0.91 (0.68-1.21)
metabolic encephalopathy	45	0.259	4.42 (3.86-5.07)	type 1 diabetes mellitus withou...	14100	-0.298	2.34 (2.03-2.70)
alzheimer's disease	46	0.256	5.03 (4.45-5.69)	open angle with borderline intr...	14101	-0.338	1.20 (1.03-1.40)
sick sinus syndrome	47	0.256	2.37 (2.08-2.71)	primary localized osteoarthrosi...	14102	-0.366	1.08 (0.82-1.43)
ventricular tachycardia	48	0.255	2.33 (2.00-2.70)	localized, primary osteoarthritis	14103	-0.393	1.23 (1.07-1.40)
acute posthemorrhagic anemia	49	0.255	2.15 (1.92-2.41)	sex_female	14104	-0.441	0.81 (0.75-0.88)
impaired fasting glycemia	50	0.254	0.97 (0.85-1.09)	open-angle glaucoma - borderline	14105	-0.495	0.97 (0.85-1.10)

8 TASK TEMPLATES

Bank Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this client subscribe to a term
deposit? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

Blood Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Did the person donate blood? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

California Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Is this house block valuable? Yes or
no?
Answer:
|||
{{ answer_choices[label] }}'
```

Car Dataset:

```
answer_choices: 'Unacceptable |||
Acceptable ||| Good ||| Very good'
jinja: '{{serialization}}

How would you rate the decision to buy
this car? Unacceptable, acceptable,
good or very good?
Answer:
|||
{{ answer_choices[label] }}'
```

Credit-g Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this person receive a credit? Yes
or no?
Answer:
|||
{{ answer_choices[label] }}'
```

Diabetes Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this patient have diabetes? Yes or
no?
Answer:
|||
{{ answer_choices[label] }}'
```

Heart Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does the coronary angiography of this
patient show a heart disease? Yes or
no?
Answer:
|||
{{ answer_choices[label] }}'
```

Income Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this person earn more than 50000
dollars per year? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

Jungle Dataset:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does the white player win this two
pieces endgame of Jungle Chess? Yes or
no?
Answer:
|||
{{ answer_choices[label] }}'
```

End Of Life Task:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this patient die in the next nine
months? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

Surgical Procedure Task:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Does this patient need a surgery in the
next nine months? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

Likelihood of Hospitalization Task:

```
answer_choices: 'No ||| Yes'
jinja: '{{serialization}}

Is this patient admitted to the hospital
in the next nine months? Yes or no?
Answer:
|||
{{ answer_choices[label] }}'
```

9 EXAMPLE SERIALIZATIONS

Bank Dataset (List Template):

```
- age:  69
- type of job:  retired
- marital status:  single
- education:  tertiary
- has credit in default?:  no
- average yearly balance, in euros:
2144
- has housing loan?:  no
- has personal loan?:  no
- contact communication type:  cellular
- last contact day of the month:  29
- last contact month of year:  jul
- last contact duration, in seconds:
417
- number of contacts performed during
this campaign and for this client:
- number of days that passed by after
the client was last contacted from a
previous campaign:  184
- number of contacts performed before
this campaign and for this client:  4
- outcome of the previous marketing
campaign:  success
```

Bank Dataset (Text Template):

```
The age is 69.  The type of job is
retired.  The marital status is single.
The education is tertiary.  The has
credit in default?  is no.  The average
yearly balance, in euros is 2144.  The
has housing loan?  is no.  The has
personal loan?  is no.  The contact
communication type is cellular.  The
last contact day of the month is 29.
The last contact month of year is jul.
The last contact duration, in seconds is
417.  The number of contacts performed
during this campaign and for this client
is.  The number of days that passed by
after the client was last contacted from
a previous campaign is 184.  The number
of contacts performed before this
campaign and for this client is 4.  The
outcome of the previous marketing
campaign is success.
```

Bank Dataset (Table-To-Text):

```
the age of 69 was 69 years.  the retired
retired.  the marital status is single
with the single name.  the school has a
school of four students.  the has a
credit of $500,000.  The average yearly
balance in euros is 2144.  the has a
total of 2,000+ housing units.  the has
an official loan of $500 million.  the
standard definition has been updated to
the standard definition.  the current
record of the month is 29.  the first
contact month was on December 20, 2005,
and then on March 22, 2006, the next
month was on March 22, 2006.  the first
contact duration was 417 seconds.  the
DVB has a selection of DVB. The year, in
which the client was first contacted by
a former airline operator, was by a
former airline operator, and by a former
airline operator, he was the first to
enter the post of the office.  the 4 is
a 4-purpose cycle.  the first of the
first 20 MB of the history history to
use the 20 MB.
```

Bank Dataset (Text T0):

```
a retired soldier shows off his tattoos.
a city is a city with a population of
singles and tertiary education.  no, the
average yearly balance is 2144 euros.
no he has no personal loan or housing
loan a man is contacting a woman on her
cell phone on the 29th day of the month.
last contact month of year was july,
last contact duration was 417 seconds.
184 days after the client was last
contacted from a previous campaign.  a
previous marketing campaign for this
client resulted in success with 4
contacts
```

Bank Dataset (Text GPT-3):

```
The person is 69 years old, retired,
single, and has a tertiary education.
They have no credit in default, and
their average yearly balance is 2144
euros.  They have no housing loan or
personal loan.  The contact
communication type is cellular, and the
last contact was on the 29th day of the
month and lasted 417 seconds.  They have
been contacted 4 times before this
campaign, and the outcome of the
previous marketing campaign was success.
```

Blood Dataset (List Template):

```
- Recency - months since last donation:
23
- Frequency - total number of donation:
1
- Monetary - total blood donated in
c.c.:  250
- Time - months since first donation:
23
```

Blood Dataset (Text Template):

```
The Recency - months since last donation
is 23.  The Frequency - total number of
donation is 1.  The Monetary - total
blood donated in c.c.  is 250.  The Time
- months since first donation is 23.
```

Blood Dataset (Table-To-Text):

```
the number of the public can be from the
number of the public.  The 1.2 has a
maximum speed of 1.2.  The first set of
the first set was in 1742 and was in
1742.
```

Blood Dataset (Text T0):

```
The donor has made 1 donation in the
last 23 months.  monetary - total blood
donated in c.c.  :  250, time - months
since first donation :  23
```

Blood Dataset (Text GPT-3):

```
The blood donor is a 23-year-old male
who has donated blood once, 250 c.c.  of
blood, 23 months ago.
```

California Dataset (List Template):

```
- median income:  3.2377
- median age:  32
- total rooms:  6597
- total bedrooms:  1579
- population:  3689
- households:  1459
- latitude:  34.15
- longitude:  -118.01
```

California Dataset (Text Template):

```
The median income is 3.2377.  The median
age is 32.  The total rooms is 6597.
The total bedrooms is 1579.  The
population is 3689.  The households is
1459.  The latitude is 34.15.  The
longitude is -118.01.
```

California Dataset (Table-To-Text):

```
there were 3.2377 people residing in the
city.  the total rooms have 6597 rooms.
the total has a total of 1579.  The
population was 3689 at the time of the
census.  The households 1459 is a
standard households.  The value 34.15 is
a value that has a value of 34.15.  The
longitude has a distance of 1.5 km and
is approximately 1.5 km.
```

California Dataset (Text T0):

```
median age of 32 years old the hotel has
a total of 6597 rooms and 1579 bedrooms.
a city has a population of 3689 and
households of 1459.  a city is located
in the southwestern part of the country
at latitude 34.15 and longitude -118.01.
```

California Dataset (Text GPT-3):

```
The house block is located in the city
of Los Angeles, in the state of
California.  The median income in the
area is $3,237, the median age is 32
years old, the total number of rooms is
6,597, the total number of bedrooms is
1,579, the population is 3,689, and the
number of households is 1,459.  The
latitude is 34.15, and the longitude is
-118.01.
```

Car Dataset (List Template):

```
- Buying price:  low
- Doors:  three
- Maintenance costs:  low
- Persons:  more than four
- Safety score:  medium
- Trunk size:  medium
```

Car Dataset (Text Template):

```
The Buying price is low.  The Doors is
three.  The Maintenance costs is low.
The Persons is more than four.  The
Safety score is medium.  The Trunk size
is medium.
```

Car Dataset (Table-To-Text):

```
The price of the price is €1,000.  the
three Doors were three.  The total
number of people in the city is more
than four.  the Safety score was 17.5.
the Trunk size is 20.5-inch.
```

Car Dataset (Text T0):

```
The refrigerator has three doors and is
very cheap.  The maintenance costs are
low for a family of more than four.  The
car has a medium safety score and a
medium trunk size.
```

Car Dataset (Text GPT-3):

```
This car a good choice for those who are
looking for a low-priced vehicle with
low maintenance costs.  It is also a
good choice for families or groups of
friends who need a car with a bit more
space than a smaller car.  The safety
score is medium, so it is not the best
choice for those who are looking for a
car with the highest safety rating.
```

Credit-g Dataset (List Template):

```
- Status of existing checking account:
0 <= ...  < 200 DM
- Duration in month:  11
- Credit history :  existing credits
paid back duly till now
- Purpose:  furniture/equipment
- Credit amount:  1577
- Savings account/bonds:  ...  >= 1000
DM
- Present employment since:  <1
- Installment rate in percentage of
disposable income:  4
- Personal status and sex:  female :
divorced/separated/married
- Other debtors / guarantors:  none
- Present residence since:  1
- Property:  real estate
- Age in years:  20
- Other installment plans:  none
- Housing:  own
- Number of existing credits at this
bank:  1
- Job:  skilled employee / official
- Number of people being liable to
provide maintenance for:  1.0
- Telephone:  none
- foreign worker:  yes
```

Credit-g Dataset (Text Template):

```
The Status of existing checking account
is 0 <= ...  < 200 DM. The Duration in
month is 11.  The Credit history is
existing credits paid back duly till
now.  The Purpose is
furniture/equipment.  The Credit amount
is 1577.  The Savings account/bonds is
...  >= 1000 DM. The Present employment
since is <1.  The Installment rate in
percentage of disposable income is 4.
The Personal status and sex is female :
divorced/separated/married.  The Other
debtors / guarantors is none.  The
Present residence since is 1.  The
Property is real estate.  The Age in
years is 20.  The Other installment
plans is none.  The Housing is own.  The
Number of existing credits at this bank
is 1.  The Job is skilled employee /
official.  The Number of people being
liable to provide maintenance for is
1.0.  The Telephone is none.  The
foreign worker is yes.
```

Credit-g Dataset (Table-To-Text):

```
the 0.2 (0.2) is a type of 00.2.  The
average annual precipitation is 11.5
millimetres (4.5 in).  the Credit
history has been paid back to a few
years.  the standard_cell is a
standard_cell.  the amount was 1577.  the
Savings account/bonds were from the
Savings account/bonds to the Savings
account/bonds.  there were 1,000
employees.  there were 4,000 people in
the city.  The male has a male score of
the female.  the debt was $12.5 million
($9.5 million in 2013).  the current
residence has a 1,000 feet (460 m) long.
the standard estate is a standard
estate.  It has a age of 20 years.  the
first installment was the first
installment in the year 2005.  The
Housing is a public transport system
that is a network of the public.  the
company has a number of existing and
existing works, and has a number of
existing and existing works.  the
company's job is job with the job name
as "Success".  the network has a network
of over 800 MT/s.  the foreign worker
has no foreign worker.
```

Credit-g Dataset (Text T0):

```
The checking account has a balance of 0
DM. A man is paying for furniture and
equipment with a credit card.  The
credit amount is 1577, the savings
account/bonds are >= 1000 DM. The
present employee has been in this job
for a year, and the installment rate is
4.  % of disposable income.  A female
who is divorced/separated/married is
requesting a loan.  The property is
located in a gated community and has
been on the market since.  The man is 20
years old and has no other installment
plans.  The number of existing credits
at this bank is 1.  A skilled employee
is liable to provide maintenance for
1.0.  A foreign worker is without a
telephone.
```

Credit-g Dataset (Text GPT-3):

```
The person is a 20-year-old female with
a checking account status of 0-200 DM.
She has been employed for less than a
year and her installment rate is 4% of
her disposable income.  She is
divorced/separated/married and has no
other debtors or guarantors.  She has
been living in her current residence for
1 year and owns real estate.  She has 1
credit at this bank and is a skilled
employee/official.  She is liable for
maintenance for 1 person.  She has no
telephone.  She is a foreign worker.
```

Diabetes Dataset (List Template):

```
- Age:  30 years
- Number of times pregnant:  1
- Diastolic blood pressure:  64 mmHg
- Triceps skin fold thickness:  32 mm
- Plasma glucose concentration at 2
hours in an oral glucose tolerance test
(GTT): 122 mg/dl
- 2-hour serum insulin:  156 µU/ml
- Body mass index:  35.1
- Diabetes pedigree function:  0.692
```

Diabetes Dataset (Text Template):

```
The Age is 30.  The Number of times
pregnant is 1.  The Diastolic blood
pressure is 64.  The Triceps skin fold
thickness is 32.  The Plasma glucose
concentration at 2 hours in an oral
glucose tolerance test (GTT) is 122.
The 2-hour serum insulin is 156.  The
Body mass index is 35.1.  The Diabetes
pedigree function is 0.692.
```

Diabetes Dataset (Table-To-Text):

```
The age was 30 years, and was the
youngest ever to enter the age.  the
number of children is 1.  The Diastolic
blood pressure is 64.  the Triceps can
run up to 32.  the 2 hours of the
glucose is 122.  the 2-hour cycle peaked
to 156.  The mass index was 35.1.  The
0.692 is a fast and pathos.
```

Diabetes Dataset (Text T0):

```
The woman is 30 years old and has been
pregnant once.  The doctor checks the
blood pressure and triceps skin fold
thickness of the patient.  The glucose
concentration at 2 hours in an oral
glucose tolerance test (GTT) was 122 and
the 2-hour serum insulin was 156.  The
pedigree function of this family is
0.692.
```

Diabetes Dataset (Text GPT-3):

```
This patient 30 years old, has been
pregnant once, has a diastolic blood
pressure of 64 mmHg, and has a triceps
skin fold thickness of 32 mm.  The
patient's plasma glucose concentration
at 2 hours in an oral glucose tolerance
test (GTT) is 122 mg/dl, and the
patient's 2-hour serum insulin is 156
µU/ml.  The patient's body mass index is
35.1, and the patient's diabetes
pedigree function is 0.692.
```

Heart Dataset (List Template):

```
- Age of the patient:  43 years
- Sex of the patient:  male
- Chest pain type:  asymptomatic
- Resting blood pressure:  132
- Serum cholesterol:  247
- Fasting blood sugar > 120 mg/dl:  yes
- Resting electrocardiogram results:
probable or definite left ventricular
hypertrophy
- Maximum heart rate achieved:  143
- Exercise-induced angina:  yes
- ST depression induced by exercise
relative to rest:  0.1
- Slope of the peak exercise ST segment:
flat
```

Heart Dataset (Text Template):

```
The Age of the patient is 43.  The Sex
of the patient is male.  The Chest pain
type is asymptomatic.  The Resting blood
pressure is 132.  The Serum cholesterol
is 247.  The Fasting blood sugar > 120
mg/dl is yes.  The Resting
electrocardiogram results is probable or
definite left ventricular hypertrophy.
The Maximum heart rate achieved is 143.
The Exercise-induced angina is yes.  The
ST depression induced by exercise
relative to rest is 0.1.  The Slope of
the peak exercise ST segment is flat.
```

Heart Dataset (Table-To-Text):

```
The male patient was the 43rd of the Age
of the patient.  The male is a male of
the same class.  The blood pressure was
132.  The Serum cave has a cave of 247.
the sugar has a low of 120 mg/dl.  the
type of the group is the type of the
group that has a group of the group.
The highest heart rate achieved is 143.
the Exercise angina has a yes value.
The ST depression has ranged from 0.1 to
0.1.  the first segment was a flat of
the ST.
```

Heart Dataset (Text T0):

```
The patient is a 43-year-old male.  The
chest pain is asymptomatic and resting
blood pressure is 132.  The doctor
checks the fasting blood sugar and finds
it is above 120 mg/dl.  The resting ECG
results showed probable or definite left
ventricular hypertrophy, with maximum
heart rate of 143 beats per minute.  The
patient had exercise-induced angina,
with ST depression induced by exercise
relative to rest of 0.1.  The slope of
the peak exercise segment is flat.
```

Heart Dataset (Text GPT-3):

```
This patient a 43-year-old male with
asymptomatic chest pain.  His resting
blood pressure is 132 mmHg and his serum
cholesterol is 247 mm/dl.  He has
fasting blood sugar > 120 mg/dl and his
resting electrocardiogram results are
probable or definite left ventricular
hypertrophy.  His maximum heart rate
achieved is 143 and he has
exercise-induced angina.  His ST
depression induced by exercise relative
to rest is 0.1 and his slope of the peak
exercise ST segment is flat.
```

Income Dataset (List Template):

```
- Age:  30
- Race:  Asian-Pac-Islander
- Sex:  Female
- Marital status:  never married
- Rel.  to head of the household:  Own
- Native country:  Taiwan
- Occupation:  execution and management
- Work class:  private sector employee
- Capital gain last year:  0
- Capital loss last year:  0
- Education:  bachelor's degree
- Work hours per week:  52
```

Income Dataset (Text Template):

```
The Age is 30.  The Race is
Asian-Pac-Islander.  The Sex is Female.
The Marital status is never married.
The Relation to head of the household is
Own-child.  The Native country is
Taiwan.  The Occupation is execution and
management.  The Work class is private
sector employee.  The Capital gain last
year is 0.  The Capital loss last year
is 0.  The Education is bachelor's
degree.  The Work hours per week is 52.
```

Income Dataset (Table-To-Text):

```
The age was 30 years, and was the
youngest ever to enter the age.  The
race was held in the Asian-Pac-Islander,
and was won by the race.  The sex of the
village was Female.  The first female to
be married is Marital status never
reported.  the family has the head of
the household.  The Chinese:  native
region of Taiwan.  He was the
executioners of the execution and
management of the city of New York City.
the private sector employee is a private
sector employee.  The capital was
Capital of the State of India.  The
capital loss of the state was 0.5%.  The
bachelor's degree in Education was
bachelor's degree.  the week 52 was the
52-hour week.
```

Income Dataset (Text T0):

```
Kim is a 30-year-old Asian-Pacific
Islander.  She is never married and has
never had children.  The man is the
owner of the house and he is the only
child.  A woman is executing a contract
as a private sector employee.  The
company had a capital loss of $ 0 last
year.  The man has a bachelor's degree
and works 52 hours a week.
```

Income Dataset (Text GPT-3):

```
The person is 30 years old,
Asian-Pac-Islander, female, never
married, and an own-child relation to
the head of the household.  The person
is from Taiwan and is an execution and
management occupation in the private
sector employee work class.  The person
has 0 dollars in capital gain and 0
dollars in capital loss from the
previous year.  The person has a
bachelor's degree and works 52 hours per
week.
```

Jungle Dataset (List Template):

```
- white piece strength:  6
- white piece file:  4
- white piece rank:  7
- black piece strength:  0
- black piece file:  5
- black piece rank:  2
```

Jungle Dataset (Text Template):

```
The white piece strength is 6.  The
white piece file is 4.  The white piece
rank is 7.  The black piece strength is
0.  The black piece file is 5.  The
black piece rank is 2.
```

Jungle Dataset (Table-To-Text):

```
the piece has a value of 6.  the 4 file
file has a 4-polytopic file.  the piece
has a cross point of the right side.
the black piece strength is 0.  The
black piece file has a 5.0.
```

Jungle Dataset (Text T0):

```
The white piece has a strength of 6 and
a file of 4.  The white piece is ranked
7, the black piece is ranked 0.  The
black piece is ranked number two.
```

Jungle Dataset (Text GPT-3):

```
The white piece is stronger than the
black piece.  The white piece is on file
4 and rank 7.  The black piece is on
file 5 and rank 2.
```

9.1 Large Healthcare Claims Dataset

End Of Life Task anonymized (List Template):

```
Summary:  The patient is a 73 year old
hispanic or latino man.

May 30, 2014:  saw a doctor for
dermatology
Conditions:
- chronic cholecystitis
- aplastic anemia due to drugs

April 21, 2017:  visited the hospital
for 12 days
Conditions:
- chronic cholecystitis [...]
```

End Of Life Task anonymized (Text Template):

```
Summary:  The patient is a 73 year old
hispanic or latino man.

On May 30, 2014 the patient saw a doctor
for dermatology with a primary complaint
of chronic cholecystitis.  He was also
treated for aplastic anemia due to
drugs.

On April 21, 2017 the patient visited
the hospital for 12 days with a primary
complaint of chronic cholecystitis.
[...]
```

End Of Life Task anonymized (List Permuted Names):

```
Summary:  The patient is a 73 year old
hispanic or latino man.

May 30, 2014:  saw a doctor for
dermatology
Conditions:
- onychomycosis due to dermatophyte
- chronic kidney disease

April 21, 2017:  visited the hospital
for 12 days
Conditions:
- onychomycosis due to dermatophyte
[...]
```

Supplementary Materials References

Avati, A., Jung, K., Harman, S., Downing, L., Ng, A., and Shah, N. H. (2018). Improving palliative care with deep learning. *BMC medical informatics and decision making*, 18(4):55–64.

Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2022). Deep Neural Networks and Tabular Data: A Survey. Technical Report arXiv:2110.01889, arXiv.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Curtis, J. R., Treece, P. D., Nielsen, E. L., Gold, J., Ciechanowski, P. S., Shannon, S. E., Khandelwal, N., Young, J. P., and Engelberg, R. A. (2016). Randomized trial of communication facilitators to reduce family distress and intensity of end-of-life care. *American journal of respiratory and critical care medicine*, 193(2):154–162.

Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J.-J., Sandhu, S., Guppy, K. H., Lee, S., and Froelicher, V. (1989). International application of a new probability algorithm for the diagnosis of coronary artery disease. *The American journal of cardiology*, 64(5):304–310.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Grinsztajn, L., Oyallon, E., and Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Hasan, M. K., Alam, M. A., Das, D., Hossain, E., and Hasan, M. (2020). Diabetes Prediction Using Ensembling of Different Machine Learning Classifiers. *IEEE Access*, 8:76516–76531. Conference Name: IEEE Access.

Hripcsak, G., Duke, J. D., Shah, N. H., Reich, C. G., Huser, V., Schuemie, M. J., Suchard, M. A., Park, R. W., Wong, I. C. K., Rijnbeek, P. R., van der Lei, J., Pratt, N., Nor&#233, N, G. N., Li, Y.-C., Stang, P. E., Madigan, D., and Ryan, P. B. (2015). Observational Health Data Sciences and Informatics (OHDSI): Opportunities for Observational Researchers. *MEDINFO 2015: eHealth-enabled Health*, pages 574–578. Publisher: IOS Press.

Kadra, A., Lindauer, M., Hutter, F., and Grabocka, J. (2021). Well-tuned simple nets excel on tabular datasets. *Advances in neural information processing systems*, 34:23928–23941.

Kohavi, R. et al. (1996). Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pages 202–207.

Moro, S., Cortez, P., and Rita, P. (2014). A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31.

Muhammad, Y., Tahir, M., Hayat, M., and Chong, K. T. (2020). Early and accurate detection and diagnosis of heart disease using intelligent computational model. *Scientific Reports*, 10(1):19747.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. *arXiv:2203.02155 [cs]*. arXiv: 2203.02155.

Pace, R. K. and Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sanh, V., Webson, A., Raffel, C., Bach, S., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Raja, A., Dey, M., Bari, M. S., Xu, C., Thakker, U., Sharma, S. S., Szczechla, E., Kim, T., Chhablani, G., Nayak, N., Datta, D., Chang, J., Jiang, M. T.-J., Wang, H., Manica, M., Shen, S., Yong, Z. X., Pandey, H., Bawden, R., Wang, T., Neeraj, T., Rozen, J., Sharma, A., Santilli, A., Fevry, T., Fries, J. A., Teehan, R., Scao, T. L., Biderman, S., Gao, L., Wolf, T., and Rush, A. M. (2022). Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Smith, J. W., Everhart, J., Dickson, W., Knowler, W., and Johannes, R. (1988). Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus. *Proceedings of the Annual Symposium on Computer Application in Medical Care*, pages 261–265.

van Rijn, J. N. and Vis, J. K. (2014). Endgame analysis of dou shou qi. *ICGA Journal*, 37(2):120–124.

Yeh, I.-C., Yang, K.-J., and Ting, T.-M. (2009). Knowledge discovery on rfm model using bernoulli sequence. *Expert Systems with Applications*, 36(3):5866–5871.