FORESTLLM: Large Language Models Make Random Forest Great on Few-shot Tabular Learning

Anonymous ACL submission

Abstract

Tabular data underpins high-stakes decision-making in finance, healthcare, and scientific discovery, yet learning effectively from it in few-shot settings remains a fundamental challenge. Decision forests are natural candidates for tabular tasks, but under limited supervision they suffer from two concrete failure modes: split selection heuristics become unstable when estimated from scarce labels, and leaf predictions degenerate into high-variance estimates based on only a handful of examples—while readily available unlabeled data is ignored entirely. Meanwhile, direct applications of large language models (LLMs) to tabular data discard the structural inductive biases that make tree-based methods effective. We propose FORESTLLM, a framework that uses an LLM to repair precisely these two bottlenecks during training, and is then removed entirely at inference. Our approach introduces (1) *semi-supervised semantic tree induction*, where the LLM evaluates candidate partitions over both labeled and unlabeled data to produce semantically grounded splits; and (2) *in-context leaf label inference*, a one-time mechanism that replaces noisy empirical estimates with domain-knowledge-informed predictions that are permanently cached. The resulting model is a lightweight, interpretable decision forest with zero LLM overhead at test time. FORESTLLM consistently matches or outperforms strong baselines across diverse few-shot classification and regression benchmarks, demonstrating the promise of targeted LLM interventions for scalable tabular learning under data scarcity.

1 Introduction

Tabular data, structured as collections of heterogeneous, feature-label pairs, is central to decision-making in critical domains such as healthcare, finance, and public policy (Hernandez et al., 2022; Frosch et al., 2010; Assefa et al., 2020; Johnson et al., 2016; Ulmer et al., 2020; Arun et al., 2016; Chen et al., 2023). However, building reliable models from such data in the few-shot regime remains a longstanding challenge. Unlike images or text, tabular data lacks spatial or sequential structure, and its highly combinatorial, non-Euclidean nature undermines the geometric assumptions underlying most modern few-shot learning techniques (Chen et al., 2018; Majumder et al., 2022; Nam et al., 2023b). As a result, these methods often perform poorly on tabular tasks with limited supervision.

Table 1: Comparison of LLM-based Tabular Methods.

Method	No LLM Fine-tuning Needed	No LLM Needed at Inference	Task Support	
			Classification	Regression
LIFT (Dinh et al., 2022)	✗	✗	✓	✓
TabLLM (Hegselmann et al., 2023)	✗	✗	✓	✗
FeatLLM (Han et al., 2024)	✓	✓	✓	✗
TP-BERTa (Yan et al., 2024)	✓	✗	✓	✓
P2T (Nam et al., 2023a)	✓	✗	✓	✓
FORESTLLM (Ours)	✓	✓	✓	✓

Tree-based models such as Random Forests and Gradient Boosted Trees (Chen and Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018) remain strong performers on tabular benchmarks, thanks to their inductive bias toward hierarchical data partitioning. Yet under low-data conditions, their reliability degrades due to two specific failure modes. First, conventional split selection heuristics, like information gain or Gini impurity, rely entirely on labeled data and become unstable when estimates are noisy, while readily available unlabeled data is ignored entirely. Second, predictions at the leaves are often based on only a handful of examples, resulting in high variance and poor calibration. These two bottlenecks—*brittle splitting* and *noisy leaf estimation*—are the core obstacles to deploying decision forests in few-shot tabular settings.

In parallel, the generalization and reasoning capabilities of Large Language Models (LLMs) have spurred interest in applying them to tabular learning (Dinh et al., 2022; Hegselmann et al., 2023; Han et al., 2024; Yan et al., 2024; Nam et al., 2024; Wen et al., 2024; Yan et al., 2025). But most exist-

ing approaches serialize rows into text and rely on prompting or fine-tuning, discarding the structural regularities that classical models are designed to exploit. Despite their flexibility, such methods frequently underperform tree ensembles on core supervised tasks, particularly in few-shot settings (Grinsztajn et al., 2022; Borisov et al., 2022; McElfresh et al., 2023; Zabërgja et al., 2024). This raises a natural question: *rather than replacing decision forests with LLMs, can we use LLMs to fix precisely the two places where few-shot forests break—while preserving their efficiency, robustness, and interpretability at test time?*

To this end, we propose ForestLLM, a framework that employs an LLM as a training-time prior to address the two failure modes identified above, and removes it entirely at inference (Table 1). To repair *brittle splitting*, we introduce *semi-supervised semantic tree induction*: each candidate split is summarized into a prompt that captures labeled and unlabeled feature distributions along with local tree context, and the LLM selects splits that are semantically plausible and aligned with the feature-space structure. To repair *noisy leaf estimation*, we propose *in-context leaf label inference*: each decision path is translated into a natural language rule, which, together with its supporting examples, is used to prompt the LLM for a prediction that is permanently cached at the leaf. Since all LLM calls are lightweight in-context API requests confined to training, ForestLLM requires no fine-tuning and incurs zero LLM overhead at inference.

We evaluate ForestLLM across a broad set of few-shot tabular benchmarks and find that it consistently matches or outperforms strong baselines across both classification and regression tasks with as few as 4 to 48 labeled examples.

Our key contributions are as follows:

- We identify two concrete failure modes of decision forests under few-shot supervision—unstable label-only splitting and high-variance leaf estimation—and show that an LLM can serve as a training-time prior to repair both, yielding a model that requires no LLM at test time.
- We propose *semi-supervised semantic tree induction*, a splitting strategy that replaces statistical heuristics with LLM-guided evaluation over both labeled and unlabeled data, regularizing tree structure through feature-space awareness.
- We develop *in-context leaf label inference*, a single-pass mechanism that replaces noisy empirical leaf estimates with domain-knowledge-informed predictions, producing deterministic, low-variance outputs without LLM inference.

2 Related Work

Few-Shot and Semi-Supervised Tabular Learning. The primary challenge in few-shot tabular learning is to build generalizable models from scarce labeled data (Chen et al., 2018; Majumder et al., 2022). A dominant strategy involves leveraging unlabeled data through semi-supervised or self-supervised pre-training to learn rich feature representations (Somepalli et al., 2021; Yoon et al., 2020; Ucar et al., 2021; Bahri et al., 2022; Wang and Sun, 2022). In contrast, ForestLLM fundamentally redefines the role of unlabeled data. Instead of using it for representation learning, our *Semi-Supervised Semantic Tree Induction* integrates feature statistics from unlabeled instances directly into the tree construction process itself. This allows unlabeled data to impose a powerful semantic prior on the model's structure, guiding the creation of decision boundaries rather than merely refining the input features.

Decision Trees and Deep Learning on Tabular Data. The performance of models on tabular data has been historically characterized by a rivalry between tree-based ensembles like GBDTs (Chen and Guestrin, 2016; Ke et al., 2017) and deep neural networks (Kadra et al., 2021; Gorishniy et al., 2021; Borisov et al., 2022). While tree models excel due to their powerful inductive bias for hierarchical partitioning, the introduction highlights their core weakness in low-data regimes: their reliance on brittle statistical splitting criteria and high-variance leaf predictions. Deep learning models, conversely, often lack the appropriate inductive biases for tabular data, leading to suboptimal performance (McElfresh et al., 2023; Zabërgja et al., 2024). ForestLLM replaces unstable statistical splits with LLM-driven semantic evaluation and stabilizes leaf predictions with our *In-Context Leaf Label Inference*, thereby retaining the structural advantages of trees while overcoming their few-shot limitations.

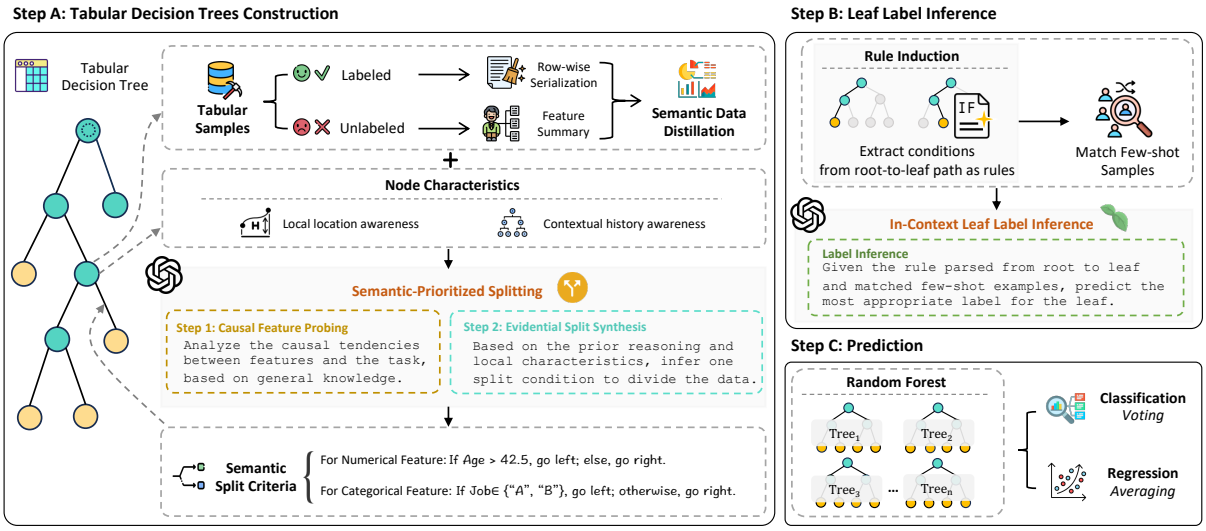

Figure 1: An overview of our proposed FORESTLLM framework for few-shot tabular learning.

Large Language Models for Tabular Tasks. Recent works have sought to apply the reasoning power of LLMs to tabular data (Dinh et al., 2022; Hegselmann et al., 2023). These models are either fine-tuned for specific tasks (Hu et al., 2021) or prompted with in-context examples for few-shot inference (Nam et al., 2023a). Prevailing approaches typically use the LLM as an online, zero-shot predictor by serializing table rows into text prompts (Nam et al., 2024), or as an offline data augmenter to generate synthetic features or examples (Han et al., 2024). However, these methods are often hampered by high inference costs, a loss of structural information during serialization, and do not produce a standalone, efficient model (Manikandan et al., 2023). FORESTLLM introduces a new paradigm, positioning the LLM as an offline semantic compiler. The LLM's knowledge is used only during training to architect a semantically coherent decision forest. Its reasoning is distilled into the tree's structure and leaves, after which the LLM is discarded. This yields a fast, interpretable model that leverages LLM intelligence without its inference-time cost or latency.

3 Method

3.1 Problem Formulation and Our Approach

We consider the problem of few-shot tabular learning, where the goal is to learn a function $h : \mathbb{R}^d \to \mathcal{Y}$ from a small labeled dataset $\mathcal{D}_L = \{(x_i, y_i)\}_{i=1}^{N_L}$ and an optional unlabeled dataset $\mathcal{D}_U = \{x_j\}_{j=1}^{N_U}$, with $N_L \ll N_U$. Here, the labeled sample size N_L corresponds to the *shot count* in our experiments. Classical decision tree methods optimize each node split s^* using statistical objectives such as information gain or Gini impurity, which rely entirely on $\mathcal{D}_L$ and often fail in low-data regimes due to high variance and weak statistical signals. In contrast, FORESTLLM constructs a forest $\mathcal{F} = \{\mathcal{T}_k\}_{k=1}^{K}$ of semantically grounded decision trees by leveraging an LLM as an *offline model designer*: the LLM guides both split selection (Section 3.2) and leaf label assignment (Section 3.3) during training, after which it is removed entirely. The resulting forest performs inference as a standard random forest with zero LLM overhead (Section 3.4). The complete algorithm is detailed in Appendix A.

3.2 Semi-Supervised Semantic Tree Induction

We propose *semantic-prioritized splitting*, a novel decision tree induction framework that leverages LLMs as a symbolic reasoner to guide recursive partitioning. Unlike classical approaches that rely solely on label-based heuristics (e.g., information gain), our method incorporates both labeled and unlabeled data, making it particularly suited for few-shot regimes. The LLM uses sparse labeled examples for supervision and dense unlabeled instances to infer semantically coherent splits aligned with observed feature distributions.

At each node n, given data $\mathcal{D}_n = \mathcal{D}_{L,n} \cup \mathcal{D}_{U,n}$, we perform *semantic data distillation* to construct a prompt for the LLM. Labeled portion $\mathcal{D}_{L,n}$ is converted into textual exemplars via *row-wise serialization*, while unlabeled portion $\mathcal{D}_{U,n}$ are summarized into a compact representation of the marginal feature distribution $P(X)$. Specifically, for each numerical feature, the summary includes descriptive statistics (count, mean, standard deviation, min-

imum, quartiles, and maximum); for each categorical feature, it reports the most frequent values and their counts, omitting rare categories to keep the prompt concise and within token limits. These components are merged with structural metadata, including the node's depth and its contextual path from the root, to form the final prompt $\Pi_{\text{prompt}} = \Psi(\mathcal{D}_{L,n}, \mathcal{D}_{U,n}) \oplus \Gamma(p_n)$. Here, Ψ is the distillation function, and $\Gamma(p_n)$ encodes the *node characteristics* that provide (1) *local depth awareness*, which calibrates split complexity, and (2) *path history awareness*, which delineates the submanifold being partitioned.

The LLM processes this prompt in two steps: (1) *Causal feature probing*, which hypothesizes relationships of the form $X_i \to Y$ based on domain knowledge and the provided data context, and (2) *Evidential split synthesis*, which translates these hypotheses into a concrete split condition s_n^* grounded in the observed feature distributions. Formally, the LLM generates the split as:

$$s_n^* = \mathcal{L}\big(\Pi_{\text{prompt}}\big)\,, \quad \Pi_{\text{prompt}} = \Psi(\mathcal{D}_{L,n}, \mathcal{D}_{U,n}) \oplus \Gamma(p_n) \tag{1}$$

where the LLM $\mathcal{L}$ jointly considers causal plausibility (whether the selected feature has a meaningful relationship with the target) and evidential support (whether the proposed threshold or category grouping is consistent with the data distribution) through its internal reasoning process. This unified generation step contrasts with classical methods that enumerate all features and thresholds: instead of exhaustive search over a combinatorial space, FORESTLLM leverages the LLM's pretrained knowledge to directly propose semantically informed splits.

Why Unlabeled Data Helps. Classical splitting criteria such as impurity reduction rely exclusively on labeled data $\mathcal{D}_{L,n}$, completely ignoring any unlabeled instances: adding $\mathcal{D}_{U,n}$ to the computation leaves the impurity score unchanged. This fundamental limitation prevents classical trees from exploiting structural information in the feature space. FORESTLLM addresses this by incorporating a *feature summary*—a compact description of the marginal feature distribution $P(X)$ estimated from the full local data $\mathcal{D}_n = \mathcal{D}_{L,n} \cup \mathcal{D}_{U,n}$—into the prompt. This enables the LLM to reason about the overall data geometry, including feature ranges, distributional modes, and correlations, when proposing splits. The design is conceptually aligned with the *cluster assumption* in semi-supervised learning (Chapelle et al., 2009), which posits that decision boundaries should pass through low-density regions of the input space. By implicitly favoring splits that respect the intrinsic data structure, FORESTLLM regularizes tree construction and mitigates overfitting to sparse labels. We empirically validate this benefit in Section 4.4, where removing unlabeled data consistently degrades performance.

3.3 In-Context Leaf Label Inference

Assuming a discrete label set $\mathcal{Y}$ for classification tasks, once a decision tree is constructed, assigning a label to a test instance reaching a leaf node ℓ requires more than conventional majority voting, which is often unreliable in few-shot or low-resource regimes. We propose *in-context leaf label inference*, a framework that reuses a pretrained LLM as a structured reasoning engine. Instead of relying on empirical aggregation over sparse leaf data, we prompt the LLM with a symbolic description of the decision path and a small set of semantically relevant exemplars, enabling context-aware prediction beyond the training distribution.

For each leaf ℓ, we extract its decision path r_ℓ from the root and translate it into a natural language rule $\mathcal{R}_\ell = \Pi_{\text{NL}}(r_\ell)$ via *Rule Induction* function. This rule captures the logic of the path as a structured, interpretable condition (e.g., "If employment $\in$ {stable, long-term} and savings $\geq$ 1000, then ..."). To ground the inference, we retrieve a support set $\mathcal{D}_L^{\text{few}} \subset \mathcal{D}_L$ consisting of labeled examples that satisfy the conditions along the decision path r_ℓ: $\mathcal{D}_L^{\text{few}} = \{(x, y) \in \mathcal{D}_L \mid x \text{ reaches } \ell\}$. We then compose a prompt $\Pi_{\text{ICL}}(\mathcal{R}_\ell, \mathcal{D}_L^{\text{few}})$ that presents the decision rule together with these supporting examples, and pass it to the LLM to produce a deterministic label:

$$\hat{y}_\ell = \arg\max_{y \in \mathcal{Y}} P_{\mathcal{L}}(y \mid \mathcal{R}_\ell, \mathcal{D}_L^{\text{few}}) \tag{2}$$

This prediction is computed once during training and permanently stored at the leaf node, ensuring that no LLM call is required at inference time. For regression tasks, the LLM produces a scalar prediction conditioned on the induced rule and supporting examples, which is directly cached at the corresponding leaf.

This design offers two key advantages over conventional leaf prediction. First, by reasoning over the full decision path as a coherent rule rather than merely counting label frequencies, the LLM can

leverage domain knowledge to produce semantically grounded predictions even when only a few (or zero) labeled examples reach the leaf. Second, because predictions are generated offline and cached, the resulting model retains the computational efficiency of a standard decision tree at test time.

3.4 Objective and Forest-Based Inference

FORESTLLM defines a semantically grounded random forest $\mathcal{F} = \{\mathcal{T}_k\}_{k=1}^{K}$, where each tree $\mathcal{T}_k$ is constructed via recursive application of the language-informed splitting procedure (Eq. 1). To promote diversity, each tree is trained on a bootstrap sample $\mathcal{D}^{(k)} \subset \mathcal{D}_L \cup \mathcal{D}_U$, and splits are selected from a random subset of features. This design preserves the statistical benefits of bagging while aligning tree structure with semantic priors induced by the LLM.

Inference on a test input x' follows a two-stage process. First, each tree $\mathcal{T}_k$ routes x' to a leaf node ℓ_k, where the static prediction $\hat{y}_k$—which is pre-computed and cached during training via *in-context leaf label inference* (Section 3.3)—is directly retrieved, incurring zero LLM overhead at test time. Second, the predictions $\{\hat{y}_1, \ldots, \hat{y}_K\}$ are aggregated into the final output $\hat{y}(x')$ using a task-specific rule: majority vote for classification or arithmetic mean for regression. By integrating semantic tree construction with offline LLM-guided label reasoning, FORESTLLM extends classical forests to the few-shot regime, achieving robust, data-efficient prediction without requiring inference-time access to the LLM.

4 Experiments

4.1 Datasets.

Following prior works (Hegselmann et al., 2023; Han et al., 2024; Abhyankar et al., 2025), we evaluate FORESTLLM on 23 tabular datasets spanning binary classification (8 datasets), multi-class classification (5 datasets), and regression (10 datasets). These datasets are drawn from OpenML, the UCI Machine Learning Repository, and Kaggle, covering diverse domains including healthcare, finance, and food. The benchmark exhibits significant diversity in both feature dimensionality (4 to 111 features) and dataset scale (315 to 253,680 samples). Full dataset descriptions, sources, and statistics are provided in Appendix B.

4.2 Baselines and Implementation Details.

To evaluate the effectiveness of our FORESTLLM, we compare against a comprehensive set of baselines spanning both LLM-based and conventional non-LLM methods.

Among LLM-based approaches, we evaluate FeatLLM (Han et al., 2024), LIFT (Dinh et al., 2022), TABLET (Slack and Singh, 2023), TabLLM (Hegselmann et al., 2023), TP-BERTa (Yan et al., 2024), and P2T (Nam et al., 2024). FORESTLLM, FeatLLM, LIFT, and TABLET use GPT-4o (gpt-4o-2024-11-20) as the base LLM, accessed via Azure. To eliminate output stochasticity, all LLM-based models are evaluated with temperature fixed to 0. Due to the high cost of fine-tuning and the small number of labeled examples, we use the in-context variant of LIFT only. In this variant, each tabular instance is serialized into natural language and the LLM is prompted to predict its label directly based on a few labeled demonstrations, without any parameter updates. This setup is functionally equivalent to direct LLM prompting on tabular data, making LIFT-ICL a representative baseline for assessing the predictive capability of the LLM itself. For TabLLM and TP-BERTa, we adopt the model configurations and training protocols reported in their original papers, using T0 and RoBERTa as respective backbones.

For traditional baselines, we include CART (Loh, 2011), Random Forest (RF)(Breiman, 2001), XGBoost(Chen and Guestrin, 2016), MLP (LeCun et al., 2015), ElasticNet (Zou and Hastie, 2005), and Logistic Regression (LogReg), along with specialized tabular learners such as TabPFN (Hollmann et al., 2025), STUNT (Nam et al., 2023b), and SCARF (Bahri et al., 2021). Notably, FeatLLM, STUNT, SCARF, and FORESTLLM leverage unlabeled data. For all such methods, unlabeled samples are consistently defined as the full dataset excluding the few-shot labeled examples and test set. All non-LLM baselines are tuned via grid search with k-fold cross-validation. Additional implementation details are provided in Appendix C.

4.3 Main Results

Tables 2 and 3 present comprehensive performance comparisons across a wide range of classification and regression tasks. For each dataset, the best-performing method is shown in bold, and the second-best is underlined. The bottom row summa-

Table 2: Classification performance (AUC) comparison of various models on 13 datasets under different few-shot settings.

Data	Shot	CART	RF	XGBoost	LogReg	SCARF	STUNT	TabPFN	TP-BERTa	TabLLM	TABLET	LIFT-ICL	FeatLLM	P2T	Ours
adult	4	0.600	0.629	0.500	0.539	0.595	0.503	0.691	0.582	0.800	0.807	0.710	**0.870**	0.765	0.833
	8	0.618	0.666	0.596	0.603	0.658	0.532	0.733	0.610	0.795	0.826	0.694	**0.880**	0.779	0.827
	16	0.639	0.725	0.685	0.667	0.658	0.541	0.746	0.658	0.821	0.819	0.667	**0.877**	0.777	0.823
blood	4	0.493	0.500	0.500	0.523	0.502	0.562	0.563	0.446	0.561	0.626	0.511	0.530	0.568	**0.691**
	8	0.542	0.575	0.567	0.564	0.617	0.553	0.604	0.473	0.578	0.648	0.551	0.598	0.618	**0.698**
	16	0.602	0.635	0.601	0.673	0.593	0.551	0.636	0.497	0.605	0.655	0.523	0.621	0.664	**0.709**
bank	4	0.535	0.613	0.500	0.585	0.556	0.531	0.602	0.401	0.609	0.829	0.616	0.723	0.683	**0.837**
	8	0.608	0.699	0.587	0.676	0.572	0.544	0.706	0.408	0.638	0.830	0.628	0.742	0.723	**0.835**
	16	0.642	0.730	0.675	0.738	0.584	0.559	0.773	0.447	0.661	0.836	0.616	0.752	0.740	**0.843**
car	4	0.562	0.554	0.500	0.580	0.571	0.564	0.612	0.585	0.558	0.840	0.799	0.703	0.656	**0.864**
	8	0.581	0.643	0.620	0.622	0.622	0.609	0.673	0.660	0.617	**0.918**	0.779	0.702	0.704	0.879
	16	0.669	0.736	0.679	0.642	0.673	0.652	0.785	0.724	0.628	0.845	0.804	0.744	0.760	**0.881**
communities	4	0.568	0.606	0.500	0.559	0.654	0.560	0.687	0.469	N/A	**0.799**	0.686	0.633	0.586	0.784
	8	0.603	0.718	0.664	0.586	0.726	0.612	0.743	0.519	N/A	**0.798**	0.725	0.691	0.594	0.782
	16	0.623	0.738	0.689	0.639	0.745	0.646	0.769	0.556	N/A	**0.794**	0.725	0.699	0.675	0.789
credit-g	4	0.515	0.526	0.500	0.521	0.539	0.508	0.532	0.467	0.580	0.623	0.506	0.498	0.538	**0.669**
	8	0.543	0.571	0.546	0.600	0.548	0.506	0.584	0.469	0.624	0.627	0.499	0.531	0.588	**0.687**
	16	0.572	0.599	0.561	0.582	0.558	0.527	0.645	0.477	0.658	0.632	0.513	0.548	0.661	**0.693**
cdc diabetes	4	0.539	0.526	0.500	0.565	0.604	0.557	0.605	0.541	0.624	**0.702**	0.655	0.604	0.570	0.700
	8	0.551	0.605	0.578	0.559	0.622	0.581	0.613	0.578	0.638	**0.703**	0.656	0.590	0.609	0.699
	16	0.562	0.611	0.600	0.590	0.600	0.572	0.630	0.602	0.620	0.701	0.649	0.633	0.599	**0.705**
heart	4	0.595	0.711	0.500	0.510	0.816	0.611	0.767	0.497	0.734	0.858	0.652	0.870	0.602	**0.887**
	8	0.639	0.721	0.594	0.659	0.867	0.594	0.828	0.521	0.809	0.858	0.563	0.880	0.763	**0.896**
	16	0.695	0.840	0.804	0.742	0.891	0.632	0.880	0.630	0.848	0.857	0.570	0.877	0.763	**0.898**
myocardial	4	0.513	0.519	0.500	0.536	0.551	0.527	0.522	0.516	N/A	0.609	0.503	0.568	0.580	**0.653**
	8	0.512	0.537	0.543	0.521	0.513	0.528	0.573	0.538	N/A	0.616	0.519	0.549	0.583	**0.660**
	16	0.547	0.592	0.569	0.537	0.519	0.538	0.608	0.587	N/A	0.632	0.570	0.577	0.576	**0.687**
breast-w	4	0.836	0.960	0.500	0.783	0.985	0.883	0.986	0.923	0.985	0.974	0.759	0.986	0.979	**0.992**
	8	0.856	0.974	0.832	0.887	0.983	0.927	0.985	0.926	0.983	0.978	0.768	0.987	0.981	**0.993**
	16	0.851	0.970	0.876	0.962	0.985	0.921	0.984	0.934	0.979	0.978	0.912	0.988	0.985	**0.993**
cultivars	4	0.514	0.493	0.500	0.504	0.494	0.514	0.499	0.445	0.524	0.525	0.531	0.528	0.536	**0.661**
	8	0.486	0.497	0.503	0.528	0.519	0.498	0.541	0.495	0.554	0.546	0.522	0.561	0.543	**0.653**
	16	0.495	0.523	0.510	0.530	0.527	0.519	0.511	0.491	0.584	0.531	0.563	0.582	0.576	**0.679**
NHANES	4	0.662	0.805	0.500	0.870	0.773	0.512	0.898	0.457	0.975	0.525	0.531	0.527	0.673	**0.987**
	8	0.962	0.900	0.880	0.968	0.812	0.518	0.976	0.519	**0.999**	0.546	0.522	0.750	0.925	0.998
	16	0.967	0.974	0.956	0.988	0.844	0.532	0.999	0.521	0.999	0.531	0.563	0.899	0.969	**0.999**
gallstone	4	0.559	0.564	0.500	0.493	0.505	0.509	0.570	0.440	0.465	0.565	0.533	0.540	0.471	**0.658**
	8	0.555	0.571	0.508	0.532	0.522	0.505	0.597	0.432	0.473	0.558	0.543	0.532	0.436	**0.653**
	16	0.592	0.591	0.589	0.596	0.537	0.528	0.612	0.426	0.461	0.563	0.529	0.557	0.481	**0.662**
Average	4	0.576	0.616	0.500	0.582	0.627	0.565	0.656	0.521	0.674	0.714	0.615	0.660	0.631	**0.786**
	8	0.620	0.667	0.617	0.639	0.660	0.577	0.704	0.550	0.701	0.727	0.613	0.692	0.680	**0.789**
	16	0.650	0.713	0.676	0.684	0.670	0.594	0.737	0.581	0.715	0.721	0.631	0.720	0.710	**0.797**

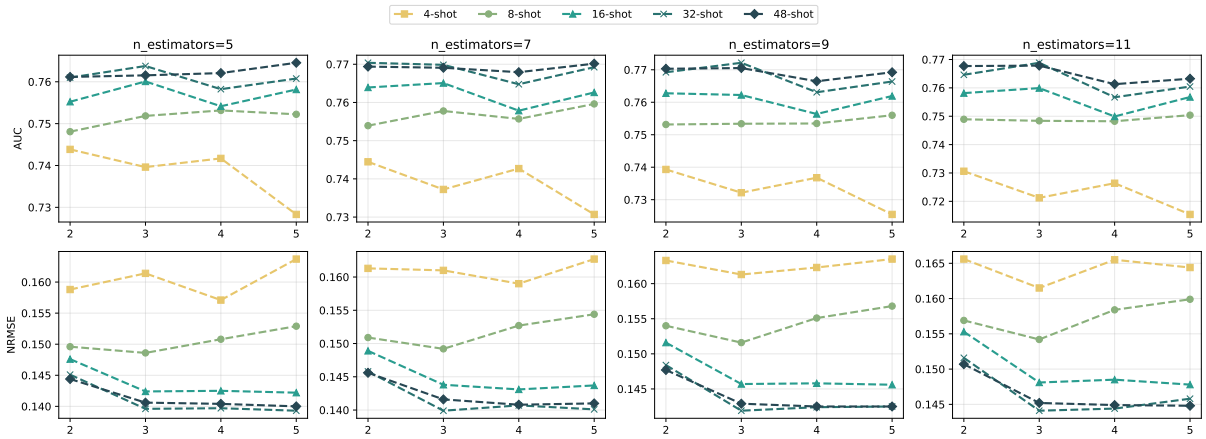

Figure 2: Hyperparameter sensitivity analysis of ForestLLM with respect to tree depth and number of estimators under varying few-shot settings. The top row shows classification results, and the bottom row shows regression results.

rizes the average performance across all datasets, visualized using a red–yellow–green gradient to indicate low, moderate, and high performance, respectively.

To account for variance introduced by train/test split randomness in few-shot scenarios, we repeat each experiment 10 times using different random seeds (0–9) for data partitioning. We report the mean AUC (for classification) and NRMSE (for regression), with standard deviations and complete performance metrics for all configurations provided in provided in Appendix G.

The results show that ForestLLM consistently achieves either the best or second-best performance across both classification and regression tasks. This superiority persists even in higher-shot settings (32- and 48-shot). Notably, ForestLLM consistently outperforms LIFT-ICL, which serves as a direct LLM prompting baseline, across nearly all datasets and shot settings. Since LIFT-ICL queries the same

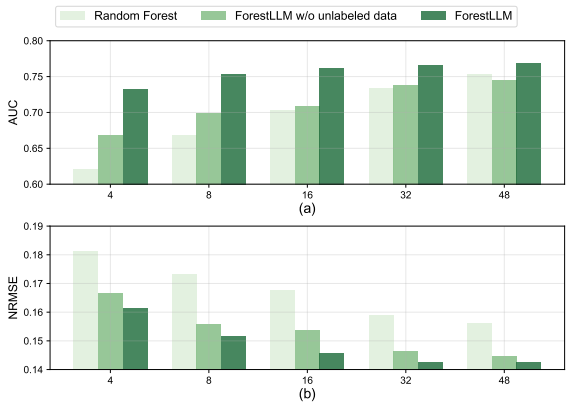

Figure 3: Ablation study of the impact of unlabeled data. (a) Classification performance measured by AUC. (b) Regression performance measured by NRMSE.

underlying LLM to predict each instance without any structural induction, this gap suggests that the decision forest architecture provides complementary inductive bias beyond the LLM's standalone reasoning capability.

We further observe that LLM-based methods tend to outperform traditional machine learning approaches on classification tasks, while traditional methods often perform better on regression. This pattern is consistent with prior work highlighting the strength of LLMs in classification tasks (Kostina et al., 2025), and their limitations in numerical reasoning and precise estimation (Drinkall et al., 2024; Siddiqui et al., 2025). FORESTLLM mitigates this limitation through a structured prediction mechanism based on random forest-style bagging and ensemble aggregation, rather than relying solely on raw LLM outputs. This design reduces uncertainty and mitigates errors introduced by the probabilistic nature of LLMs in numerical prediction tasks.

Beyond predictive performance, we conduct an extensive analysis of the training efficiency and economic cost associated with LLM API usage in Appendix D, demonstrating that FORESTLLM is a cost-effective solution for few-shot tabular learning.

4.4 Ablations

Unlabeled Data Helps. We perform an ablation study to assess the contribution of unlabeled data to FORESTLLM's performance. As shown in Figure 3, we compare FORESTLLM against (i) a variant trained using labeled data only, and (ii) a traditional Random Forest, under both classification and regression tasks. To ensure fair comparison, we adapt tree depth to the number of labeled examples. Specifically, for 4-, 8-, and 16-shot settings, we use `max_depth=3` and `n_estimators=9`; for 32- and 48-shot settings, we increase the depth to `max_depth=5`, keeping `n_estimators=9`. This adjustment accounts for the fact that, under extreme data scarcity, the limited number of labeled examples constrains tree growth depth, and shallow structures also help mitigate potential overfitting risks. In contrast, FORESTLLM is capable of leveraging unlabeled data to construct deeper, semantically guided trees even in low-shot regimes, rendering it less sensitive to the number of labeled instances. The results show that FORESTLLM consistently outperforms both baselines across all settings, confirming the effectiveness of utilizing unlabeled data for improved generalization.

Hyperparameters. We also evaluate the robustness of FORESTLLM with respect to its key hyperparameters. As shown in Figure 2, we conduct a grid search over `max_depth` and `n_estimators`, reporting both AUC and NRMSE across 4-, 8-, 16-, 32-, and 48-shot settings. We vary `max_depth` from 2 to 5 and sweep `n_estimators` over 5, 7, 9, 11. Results indicate that FORESTLLM maintains strong performance across a wide range of configurations. While deeper trees and larger ensembles can offer marginal gains, the model is notably stable, especially in higher-shot settings (e.g., 32- and 48-shot). This robustness suggests that FORESTLLM performs reliably without extensive hyperparameter tuning, making it a practical choice for real-world few-shot tabular learning.

Furthermore, we evaluate the sensitivity of FORESTLLM to different LLM backbones, including GPT-4-32k and GPT-3.5-Turbo; these results are detailed in Appendix F and suggest that our framework maintains robust performance across various foundation models.

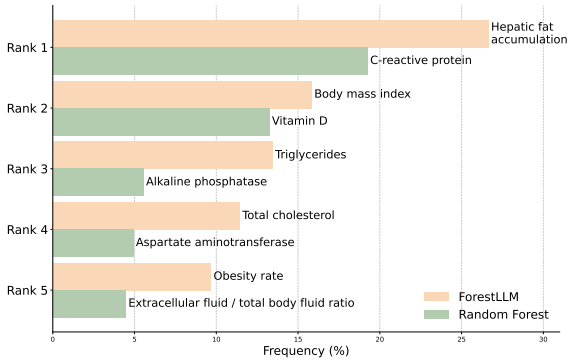

Figure 4: Comparison of the top 5 most frequently split features on the Gallstone dataset.

Semantic Reasoning vs. Statistical Splitting. To

Table 3: Regression performance (NRMSE) comparison of various models on 10 datasets under different few-shot settings. * refers to `infrared_thermography_temperature`.

Data	Shot	CART	RF	XGBoost	ElasticNet	MLP	TabPFN	TP-BERTa	LIFT-ICL	P2T	Ours
bike	4	0.220	0.194	0.224	0.266	0.225	0.192	0.264	0.161	0.190	**0.156**
	8	0.300	0.197	0.243	0.211	0.247	0.191	0.262	0.259	0.185	**0.149**
	16	0.304	0.200	0.193	0.202	0.256	0.165	0.261	0.156	0.194	**0.144**
cpu_small	4	0.168	0.180	0.198	0.322	49.042	0.167	0.789	0.173	0.234	**0.161**
	8	0.172	0.167	0.187	0.389	9.865	0.164	0.770	0.172	0.167	**0.154**
	16	0.156	0.122	0.180	0.335	0.987	0.118	0.745	**0.101**	0.147	0.114
diamonds	4	0.152	0.152	0.156	0.142	0.153	0.157	0.302	**0.080**	0.116	0.090
	8	0.150	0.118	0.129	0.169	0.116	0.153	0.302	**0.080**	0.109	0.095
	16	0.126	0.120	0.134	0.123	0.115	0.119	0.302	**0.082**	0.109	0.095
forest-fires	4	0.127	0.127	0.127	0.128	0.130	0.127	0.128	0.127	0.127	**0.126**
	8	0.204	0.143	0.140	0.170	0.128	0.170	0.127	0.153	0.136	**0.126**
	16	0.214	0.146	0.144	0.147	0.133	0.136	0.126	0.158	0.143	**0.123**
houses	4	0.252	0.180	0.156	0.393	6.056	0.176	1.270	0.214	0.358	**0.150**
	8	0.181	0.170	0.161	0.286	6.218	0.142	0.801	0.169	0.192	**0.131**
	16	0.175	0.150	0.156	0.183	2.840	0.131	0.285	0.148	0.163	**0.128**
insurance	4	0.282	0.236	0.236	0.240	0.235	0.194	0.325	0.125	0.186	**0.115**
	8	0.295	0.255	0.245	0.224	0.207	0.163	0.325	0.116	0.145	**0.107**
	16	0.230	0.232	0.234	0.185	0.193	0.158	0.325	0.116	0.119	**0.105**
retinol	4	0.292	0.239	**0.215**	0.490	0.396	0.227	0.633	0.331	0.412	0.216
	8	0.304	0.240	0.234	0.388	0.472	0.242	0.631	0.340	0.316	**0.203**
	16	0.296	0.217	0.220	0.301	0.445	0.203	0.628	0.340	0.268	**0.192**
wine	4	0.209	0.170	0.169	0.206	0.361	0.166	0.150	0.164	0.252	**0.141**
	8	0.188	0.167	0.161	0.193	0.228	0.153	0.146	0.158	0.183	**0.137**
	16	0.187	0.156	0.158	0.178	0.207	0.155	0.148	0.158	0.161	**0.132**
cultivars	4	0.268	0.234	0.218	0.325	0.314	0.300	1.403	0.370	0.328	**0.206**
	8	0.263	0.235	0.224	0.369	0.330	0.221	1.402	0.321	0.288	**0.196**
	16	0.277	0.234	0.231	0.224	0.369	0.217	1.400	0.270	0.266	**0.198**
infrared_t_t*	4	0.130	0.116	0.151	0.152	0.288	0.138	6.628	0.129	1.637	**0.108**
	8	0.129	0.109	0.160	0.132	0.138	0.110	5.898	0.105	0.722	**0.102**
	16	0.117	0.099	0.109	0.120	0.140	0.112	5.036	**0.098**	0.106	0.103
Average	4	0.210	0.183	0.185	0.266	5.720	0.184	1.189	0.187	0.384	**0.147**
	8	0.219	0.180	0.188	0.253	1.795	0.171	1.066	0.187	0.244	**0.140**
	16	0.208	0.168	0.176	0.200	0.569	0.151	0.926	0.163	0.168	**0.133**

assess interpretability in feature selection, we compared FORESTLLM (under a 4-shot setup) with a fully supervised Random Forest on the Gallstone dataset. Feature importance was quantified via split frequency, revealing distinct selection patterns between the two models. FORESTLLM prioritized upstream, causally relevant factors, such as hepatic fat, BMI, and cholesterol, consistent with known metabolic pathways implicated in gallstone formation (Portincasa et al., 2023; Wu et al., 2024; Tang et al., 2025; Deng et al., 2025). In contrast, Random Forest emphasized downstream biomarkers including CRP, ALP, and AST, which are more reflective of inflammation and hepatic injury (Zhou et al., 2021; Esen et al., 2024; Van Nynatten et al., 2025).

This divergence illustrates two fundamentally different paradigms: FORESTLLM leverages semantic priors and causal abstractions derived from the LLM to surface clinically meaningful features, even in sparse data regimes. Random Forest, by contrast, relies on purely statistical associations, which often highlight features correlated with disease severity rather than etiology (Esen et al., 2024; Van Nynatten et al., 2025). These results suggest that FORESTLLM provides a causally grounded and interpretable alternative to conventional correlation-driven models, particularly in few-shot settings. Prior studies have highlighted the value of causal feature selection for improving interpretability and clinical insight (Chen et al., 2022; Malec et al., 2023). To provide a more granular view of the semantic reasoning traces and the resulting decision structures, we provide additional case studies in Appendix E.

5 Conclusion

We identified two concrete failure modes that prevent decision forests from generalizing in few-shot tabular settings—unstable label-only splitting and high-variance leaf estimation—and demonstrated that both can be repaired through targeted LLM interventions confined entirely to the training phase. The resulting FORESTLLM framework preserves the interpretability and inference-time autonomy of classical forests while achieving robust generalization under extreme data scarcity. Our extensive benchmarks confirm that the forest's structural inductive bias and the LLM's semantic reasoning are fundamentally complementary: FORESTLLM consistently outperforms direct LLM prompting, suggesting that structural constraints are essential for grounding semantic insights. By repositioning LLMs as minimal, training-time priors rather than inference-time oracles, our work advocates for a broader design principle: intervene only where the classical model fails, and compile the acquired knowledge into a standalone, deployable learner.

6 Limitations

FORESTLLM's semantic splitting relies on the assumption that feature names carry interpretable, domain-relevant semantics. Our benchmarks benefit from descriptive identifiers (e.g., *employment*, *hepatic fat*); when features are anonymized or encoded (e.g., *V1*, *feature_032*) due to privacy constraints, the LLM cannot perform meaningful causal reasoning and split quality may degrade. Relatedly, the framework assumes that the LLM possesses sufficient domain knowledge for the target task. Our evaluation covers domains well-represented in LLM pretraining corpora (finance, healthcare, system performance); for highly specialized or niche domains, the semantic reasoning driving split selection and leaf label inference may become unreliable, and since each split is generated via a single LLM call without statistical validation or backtracking, such errors could propagate along the tree.

Finally, our experiments are conducted with GPT series models, with results in Appendix F demonstrating robustness across different model variants. We have not evaluated open-source LLMs (e.g., LLaMA, Qwen), as they differ in instruction-following capability; directly transferring our prompting strategy may not fully elicit their potential, and tailored prompt engineering is left to future work. Nonetheless, this work provides a viable path for integrating decision tree structural inductive biases with LLM-driven semantic reasoning, and we expect that exploring a broader range of LLM backbones will further validate the generality of this direction.

References

Nikhil Abhyankar, Parshin Shojaee, and Chandan K Reddy. 2025. Llm-fe: Automated feature engineering for tabular data with llms as evolutionary optimizers. *arXiv preprint arXiv:2503.14434*.

Kumar Arun, Garg Ishan, and Kaur Sanmeet. 2016. Loan approval prediction based on machine learning approach. *IOSR J. Comput. Eng*, 18(3):18–21.

Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. 2020. Generating synthetic data in finance: opportunities, challenges and pitfalls. *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8.

Arthur Asuncion, David Newman, and 1 others. 2007. Uci machine learning repository.

Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. 2021. Scarf: Self-supervised contrastive learning using random feature corruption. *arXiv preprint arXiv:2106.15147*.

Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. 2022. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*.

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G. Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. 2019. Openml benchmarking suites. *arXiv:1708.03731v2 [stat.ML]*.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2022. Deep neural networks and tabular data: A survey. *IEEE transactions on neural networks and learning systems*.

Leo Breiman. 2001. Random forests. *Machine learning*, 45:5–32.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Tingting Chen, Vignesh Sampath, Marvin Carl May, Shuo Shan, Oliver Jonas Jorg, Juan José Aguilar Martín, Florian Stamer, Gualtiero Fantoni, Guido Tosello, and Matteo Calaon. 2023. Machine learning in manufacturing towards industry 4.0: From 'for now'to 'four-know'. *Applied Sciences*, 13(3):1903.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. 2018. A closer look at few-shot classification. In *International Conference on Learning Representations*.

YuWen Chen, Ju Zhang, and XiaoLin Qin. 2022. Interpretable instance disease prediction based on causal feature selection and effect analysis. *BMC medical informatics and decision making*, 22(1):51.

Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. 2009. Modeling wine preferences by data mining from physicochemical properties. *Decision support systems*, 47(4):547–553.

Bruno Rodrigues de Oliveira, Alan Mario Zuffo, Francisco Charles dos Santos Silva, Ricardo Mezzomo, Leandra Matos Barrozo, Tatiane Scilewski da Costa Zanatta, Joel Cabral dos Santos, Carlos Henrique Conceição Sousa, and Yago Pinto Coelho. 2023. Dataset: Forty soybean cultivars from subsequent harvests. *Trends in Agricultural and Environmental Sciences*, pages e230005–e230005.

Laifu Deng, Shuting Wang, Daiwei Wan, Qi Zhang, Wei Shen, Xiao Liu, and Yu Zhang. 2025. Relative fat mass and physical indices as predictors of gallstone formation: insights from machine learning and logistic regression. *International Journal of General Medicine*, pages 509–527.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. 2022. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784.

Felix Drinkall, Janet B Pierrehumbert, and Stefan Zohren. 2024. Forecasting credit ratings: A case study where traditional methods outperform generative llms. *arXiv preprint arXiv:2407.17624*.

İrfan Esen, Hilal Arslan, Selin Aktürk Esen, Mervenur Gülşen, Nimet Kültekin, and Oğuzhan Özdemir. 2024. Early prediction of gallstone disease with a machine learning-based method from bioimpedance and laboratory data. *Medicine*, 103(8):e37258.

Hadi Fanaee-T. 2013. Bike Sharing. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5W894.

Sebastian Felix Fischer, Matthias Feurer, and Bernd Bischl. 2023. Openml-ctr23–a curated tabular regression benchmarking suite. In *AutoML Conference 2023 (Workshop)*.

Dominick L Frosch, David Grande, Derjung M Tarn, and Richard L Kravitz. 2010. A decade of controversy: balancing policy with evidence in the regulation of prescription drug advertising. *American Journal of Public Health*, 100(1):24–32.

SE Golovenkin, VA Shulman, DA Rossiev, PA Shesternya, S Yu Nikulina, Yu V Orlova, and VF Voino-Yasenetsky. 2020. Myocardial infarction complications. uci machine learning repository. *UCI Machine Learning Repository*.

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

Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. 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*.

Sungwon Han, Jinsung Yoon, Sercan O Arik, and Tomas Pfister. 2024. Large language models can automatically engineer features for few-shot tabular learning. In *International Conference on Machine Learning*, pages 17454–17479. PMLR.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR.

Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. 2022. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 493:28–45.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li wei H. Lehman, Mengling Feng, Mohammad Mahdi Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific Data*, 3.

Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka. 2021. Well-tuned simple nets excel on tabular datasets. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Arina Kostina, Marios D Dikaiakos, Dimosthenis Stefanidis, and George Pallis. 2025. Large language models for text classification: Case study and comprehensive review. *arXiv preprint arXiv:2501.08457*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Wei-Yin Loh. 2011. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(1):14–23.

Sagnik Majumder, Changan Chen, Ziad Al-Halah, and Kristen Grauman. 2022. Few-shot audio-visual learning of environment acoustics. *Advances in Neural Information Processing Systems*, 35:2522–2536.

Scott A Malec, Sanya B Taneja, Steven M Albert, C Elizabeth Shaaban, Helmet T Karim, Arthur S Levine, Paul Munro, Tiffany J Callahan, and Richard D Boyce. 2023. Causal feature selection using a knowledge graph combining structured knowledge from the biomedical literature and ontologies: a use case studying depression as a risk factor for alzheimer's disease. *Journal of biomedical informatics*, 142:104368.

Hariharan Manikandan, Yiding Jiang, and J Zico Kolter. 2023. Language models are weak learners. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Ganesh Ramakrishnan, Vishak Prasad, Micah Goldblum, and Colin White. 2023. When do neural nets outperform boosted trees on tabular data? In *Advances in Neural Information Processing Systems*.

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

Jaehyun Nam, Woomin Song, Seong Hyeon Park, Jihoon Tack, Sukmin Yun, Jaehyung Kim, Kyu Hwan Oh, and Jinwoo Shin. 2024. Tabular transfer learning via prompting LLMs. In *First Conference on Language Modeling*.

Jaehyun Nam, Woomin Song, Seong Hyeon Park, Jihoon Tack, Sukmin Yun, Jaehyung Kim, and Jinwoo Shin. 2023a. Semi-supervised tabular classification via in-context learning of large language models. In *Workshop on Efficient Systems for Foundation Models@ ICML2023*.

Jaehyun Nam, Jihoon Tack, Kyungmin Lee, Hankook Lee, and Jinwoo Shin. 2023b. Stunt: Few-shot tabular learning with self-generated tasks from unlabeled tables. In *The Eleventh International Conference on Learning Representations*.

Piero Portincasa, Agostino Di Ciaula, Leonilde Bonfrate, Alessandro Stella, Gabriella Garruti, and John Thomas Lamont. 2023. Metabolic dysfunction-associated gallstone disease: expecting more from critical care manifestations. *Internal and emergency medicine*, 18(7):1897–1918.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31.

Michael Redmond. 2009. Communities and crime. uci machine learning repository.

Shoaib Ahmed Siddiqui, Yanzhi Chen, Juyeon Heo, Menglin Xia, and Adrian Weller. 2025. On evaluating LLMs' capabilities as functional approximators: A Bayesian evaluation framework. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5826–5835, Abu Dhabi, UAE. Association for Computational Linguistics.

Dylan Slack and Sameer Singh. 2023. Tablet: Learning from instructions for tabular data. *arXiv preprint arXiv:2304.13188*.

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

Yong Feng Tang, Yong Tian Su, Li Juan Liang, Yong Feng, Xiang Jiao Huang, Xue Lian Xiang, and Zhi Hai Liang. 2025. Association between metabolic dysfunction and gallstone disease in us adults: An analysis of the national health and nutrition examination survey. *Journal of Digestive Diseases*.

Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. 2021. Subtab: Subsetting features of tabular data for self-supervised representation learning. *Advances in Neural Information Processing Systems*, 34:18853–18865.

Dennis Ulmer, Lotta Meijerink, and Giovanni Cinà. 2020. Trust issues: Uncertainty estimation does not enable reliable ood detection on medical tabular data. In *Machine Learning for Health*, pages 341–354. PMLR.

Logan R Van Nynatten, Maitray A Patel, Mark Daley, Michael R Miller, Gediminas Cepinskas, Marat Slessarev, James A Russell, and Douglas D Fraser. 2025. Putative biomarkers of hepatic dysfunction in critically ill sepsis patients. *Clinical and Experimental Medicine*, 25(1):28.

Quanzeng Wang, Yangling Zhou, Pejman Ghassemi, Dwith Chenna, Michelle Chen, Jon Casamento, Joshua Pfefer, and David Mcbride. 2023. Facial and oral temperature data from a large set of human subject volunteers. *PhysioNet, May*.

Zifeng Wang and Jimeng Sun. 2022. Transtab: Learning transferable tabular transformers across tables. *Advances in Neural Information Processing Systems*, 35:2902–2915.

Xumeng Wen, Han Zhang, Shun Zheng, Wei Xu, and Jiang Bian. 2024. From supervised to generative: A novel paradigm for tabular deep learning with large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3323–3333.

Weigen Wu, Yuchen Pei, Junlong Wang, Qizhi Liang, and Wei Chen. 2024. Association between visceral lipid accumulation indicators and gallstones: a cross-sectional study based on nhanes 2017–2020. *Lipids in Health and Disease*, 23(1):345.

Jiahuan Yan, Jintai Chen, Chaowen Hu, Bo Zheng, Yaojun Hu, Jimeng Sun, and Jian Wu. 2025. Small models are LLM knowledge triggers for medical tabular prediction. In *The Thirteenth International Conference on Learning Representations*.

Jiahuan Yan, Bo Zheng, Hongxia Xu, Yiheng Zhu, Danny Chen, Jimeng Sun, Jian Wu, and Jintai Chen. 2024. Making pre-trained language models great on tabular prediction. In *The Twelfth International Conference on Learning Representations*.

I-Cheng Yeh, King-Jang Yang, and Tao-Ming Ting. 2009. Knowledge discovery on rfm model using bernoulli sequence. *Expert Systems with applications*, 36(3):5866–5871.

Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. 2020. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33:11033–11043.

G Zabërgja, A Kadra, and J Grabocka. 2024. Is deep learning finally better than decision trees on tabular data. *arXiv preprint arXiv:2402.03970*.

Ji-Yuan Zhou, Liu-Wei Song, Rong Yuan, Xiao-Ping Lu, and Gui-Qiang Wang. 2021. Prediction of hepatic inflammation in chronic hepatitis b patients with a random forest-backward feature elimination algorithm. *World journal of gastroenterology*, 27(21):2910.

Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320.

A Algorithm

Algorithm 1 provides a complete summary of the FORESTLLM training and inference procedure. The offline training phase uses the LLM to construct decision trees and assign static leaf predictions. The online inference phase operates entirely as a standard random forest.

B Dataset Details

We evaluate our method on a diverse suite of tabular datasets drawn from popular benchmarks: OpenML-CC18 (Bischl et al., 2019) for classification, OpenML-CTR23 (Fischer et al., 2023) for regression, and additional datasets from the UCI Machine Learning Repository and Kaggle. Table 4 provides a summary organized by task type: binary classification, multi-class classification, and regression. For each dataset, we report key statistics, including the number of samples, number of features, label distribution, and dataset identifier. These datasets span a wide range of domains, scales, and structural properties, providing a rigorous testbed for assessing generalization and robustness in few-shot tabular learning.

Specifically, for *binary classification*, we include Adult (Asuncion et al., 2007), Bank (Yeh et al., 2009), Blood (Moro et al., 2014), Credit-g (Kadra et al., 2021), Cultivars (de Oliveira et al., 2023), Gallstone (Esen et al., 2024), and NHANES.[1] For *multi-class classification*, we use Car (Kadra et al., 2021), CDC Diabetes,[2] Heart,[3] Communities (Redmond, 2009), and Myocardial (Golovenkin et al., 2020). For *regression*, we evaluate on Cultivars, Cpu_small,[4] Diamonds,[5] Plasma_retinol,[6] Forest-fires,[7] Housing,[8] Insurance,[9] Bike (Fanaee-T, 2013), Wine (Cortez et al., 2009), and Infrared_Thermography_Temperature (Wang et al., 2023).

[1] https://archive.ics.uci.edu/dataset/887
[2] https://kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset
[3] https://kaggle.com/datasets/fedesoriano/heart-failure-prediction
[4] https://openml.org/search?type=data&id=562
[5] https://openml.org/search?type=data&id=42225
[6] https://openml.org/search?type=data&id=511
[7] https://openml.org/search?type=data&id=42363
[8] https://openml.org/search?type=data&id=43996
[9] https://kaggle.com/datasets/teertha/ushealthinsurancedataset

C Implementation Details

Data Preparation. Following the popular setting (Han et al., 2024), we partition each dataset into training and test sets, reserving 20% of the data for testing. For classification tasks, we apply stratified sampling to preserve the label distribution across classes. From the training set, we construct a k-shot subset for supervision, selecting examples in a class-balanced manner whenever possible. In scenarios with limited data or many classes, we permit approximate balancing to maintain representativeness. For regression tasks, we discretize the continuous response variable into quantile-based bins and sample within each bin to maintain the marginal target distribution. To account for variability due to stochastic sampling, especially prominent in few-shot regimes, we repeat each experiment using 10 different random seeds (0 through 9) and report mean performance metrics. All methods receive the same train/test partitions under each seed to ensure consistency and fairness. To minimize confounding effects introduced by preprocessing, we adopt a lightweight and uniform imputation strategy. For LLM-based methods, all missing values—whether numerical or categorical—are replaced with the string "`Unknown`", which serves as a symbolic placeholder compatible with text prompts. For classical models, we impute categorical variables with "`Unknown`" and continuous variables with the column-wise mean. This design ensures that preprocessing does not bias the evaluation of semantic modeling capabilities or statistical performance across methods.

Hyperparameter Tuning. Hyper-parameter selection is essential for fair comparison, particularly for classical machine learning baselines such as Logistic Regression (LogReg), Random Forest (RF), XGBoost, ElasticNet, Multilayer Perceptron (MLP), and TabPFN. We perform hyperparameter tuning for these models using grid search combined with k-shot cross-validation on the training data. Specific hyperparameter grids are detailed in Tables 5–11. For FORESTLLM, we perform hyperparameter tuning via grid search without cross-validation to minimize API costs. In the few-shot regime, conventional train/validation splits (e.g., 70%/30%) yield validation sets too small for stable evaluation. Instead, we evaluate each configuration directly on the training set, which we find to be a practical compromise under severe data constraints. The hyperparameter search space is summarized in Ta-

Algorithm 1 FORESTLLM: Training and Inference

Input: Labeled data $\mathcal{D}_L$, unlabeled data $\mathcal{D}_U$, number of trees K, max depth D, task description τ
Output: Trained forest $\mathcal{F} = \{\mathcal{T}_k\}_{k=1}^{K}$ with static leaf predictions

% —- Offline Training (uses LLM) —-
for $k = 1, \ldots, K$ **do**
  $\mathcal{D}^{(k)} \leftarrow$ BOOTSTRAP$(\mathcal{D}_L \cup \mathcal{D}_U)$
  $\mathcal{D}_L^{(k)} \leftarrow \mathcal{D}^{(k)} \cap \mathcal{D}_L$ ▷ Labeled part of bootstrap
  $\mathcal{T}_k \leftarrow$ BUILDTREE$(\mathcal{D}^{(k)}, D, \tau)$ ▷ Stage A
  for each leaf ℓ in $\mathcal{T}_k$ **do** ▷ Stage B
    $\mathcal{R}_\ell \leftarrow$ extract root-to-leaf path as natural language rule
    $\mathcal{D}_L^{(k),\text{few}} \leftarrow$ labeled samples in $\mathcal{D}_L^{(k)}$ satisfying $\mathcal{R}_\ell$
    $\hat{y}_\ell \leftarrow \text{LLM}(\tau,\ \mathcal{R}_\ell,\ \mathcal{D}_L^{(k),\text{few}})$ ▷ One-time call
    store $\hat{y}_\ell$ in leaf ℓ ▷ Static prediction
  end for
end for
return $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$

% —- Online Inference (no LLM) —-
function PREDICT$(\mathbf{x}', \mathcal{F})$
  for $k = 1, \ldots, K$ **do**
    $\ell_k \leftarrow$ route $\mathbf{x}'$ through $\mathcal{T}_k$ via stored split conditions
    $\hat{y}_k \leftarrow$ **retrieve** stored $\hat{y}_{\ell_k}$ ▷ No LLM call
  end for
  return AGG$(\hat{y}_1, \ldots, \hat{y}_K)$ ▷ Vote or average
end function

% —- Recursive Tree Construction —-
function BUILDTREE$(\mathcal{D}_n, D, \tau, \text{depth} = 0)$
  if depth $= D$ **or** $|\mathcal{D}_{L,n}| <$ min_samples **or** $\mathcal{D}_{L,n}$ is pure **then**
    return leaf node ▷ Label assigned in Stage B
  end if
  $\mathcal{F}_{\text{sub}} \leftarrow$ random subset of features
  prompt $\leftarrow$ compose(τ, serialize($\mathcal{D}_{L,n}$), summarize($\mathcal{D}_n$, $\mathcal{F}_{\text{sub}}$), node_context)
  $(f, v) \leftarrow \text{LLM}(\textit{prompt})$ ▷ Split: feature f, value v
  $\mathcal{D}_n^L, \mathcal{D}_n^R \leftarrow$ partition $\mathcal{D}_n$ by condition (f, v)
  left $\leftarrow$ BUILDTREE$(\mathcal{D}_n^L, D, \tau, \text{depth} + 1)$
  right $\leftarrow$ BUILDTREE$(\mathcal{D}_n^R, D, \tau, \text{depth} + 1)$
  return internal node with split (f, v), children left, right
end function

Table 4: Dataset Characteristics

Dataset	#Samples	#Features	#Label ratio(%)	Source	ID/Name
Binary Classification					
adult	48842	14	76:24	OpenML	1590
bank	45211	16	88:12	OpenML	1461
blood	748	4	76:24	OpenML	1464
breast-w	699	9	66:34	OpenML	15
credit-g	1000	20	70:30	OpenML	31
cultivars	320	10	50:50	UCI	Forty Soybean Cultivars from Subsequent Harvests
gallstone	320	38	50:50	UCI	Gallstone
NHANES	6287	8	84:16	UCI	National Health and Nutrition Health Survey 2013-2014 Age Prediction Subset
Multi-class Classification					
car	1728	6	70:22:4:4	OpenML	40975
communities	1994	103	34:33:33	UCI	Communities and Crime
myocardial	1700	111	22:78	UCI	Myocardial infarction complications
heart	918	11	45:55	Kaggle	Heart Failure Prediction Dataset
cdc diabetes	253680	21	84:14:2	Kaggle	Diabetes Health Indicators Dataset
Regression					
Cpu_small	8192	12	N/A	OpenML	562
diamonds	53940	19	N/A	OpenML	42225
forest-fires	517	13	N/A	OpenML	42363
housing	20640	9	N/A	OpenML	43996
plasma_retinol	315	13	N/A	OpenML	511
bike	17389	11	N/A	UCI	Bike Sharing
cultivars	320	10	N/A	UCI	Forty Soybean Cultivars from Subsequent Harvests
infrared_thermography_temperature	1020	33	N/A	UCI	Infrared Thermography Temperature
wine	4898	10	N/A	UCI	Wine Quality
insurance	1338	7	N/A	Kaggle	US Health Insurance Dataset

ble 12.

Evaluation. Classification performance is reported using the macro-averaged Area Under the ROC Curve (AUC), computed via the `roc_auc_score` function from `sklearn.metrics`. For LLM-based baselines including LIFT-ICL, TABLET, and P2T, prediction confidences are derived from token-level log-probabilities returned by the OpenAI GPT-4o API, following best practices for prompt-based classification. For regression, we report normalized root mean squared error (NRMSE), with normalization based on the standard deviation of the target variable in the test set. All reported metrics are averaged over 10 independent trials with different train/test seeds. Experiments are executed on a dedicated Ubuntu 22.04.4 LTS server equipped with an AMD EPYC 7V13 64-core CPU, 432 GB of RAM, and two NVIDIA A100 80GB GPUs. OpenAI GPT-4o (engine: `gpt-4o-2024-11-20`) is accessed via the Azure OpenAI API. All LLM-based baselines (LIFT-ICL, TABLET, TabLLM, FeatLLM, P2T) issue a single prediction per instance, so we set their temperature to 0 to eliminate sampling stochasticity. For ForestLLM, we likewise use temperature 0 during leaf label inference, ensuring that cached leaf predictions are deterministic. During tree construction, we instead use temperature 0.5, which plays the same role as the bootstrap and random feature subsampling in classical random forests by encouraging different trees to explore different splits from the same node-level prompt. Because each tree is built independently and the LLM is invoked only at training time, this stochasticity is fully absorbed into the trained forest and never affects inference. The reported standard deviations therefore capture the end-to-end variance of training and evaluation, consistent with how randomness is reported for tree-based baselines whose bootstrap and feature sampling are similarly folded into their std.

Robustness Against LLM Hallucinations. Since FORESTLLM relies on LLM-generated split conditions, we implement a multi-level verification pipeline to guard against hallucinated or invalid outputs.

Split-level verification. Each LLM-proposed split undergoes three checks before acceptance: (1) the output is constrained to a structured JSON schema containing the selected feature, threshold (for numerical features), or category grouping (for categorical features), reducing format-level errors; (2) the proposed feature must exist in the current candidate feature subset—splits referencing fabricated features are discarded; and (3) the resulting partition must satisfy a minimum samples-per-child constraint—if the proposed threshold produces an extremely imbalanced split (e.g., all data on one side), the split is rejected and the node becomes a leaf.

Leaf-level verification. For classification tasks, the LLM-predicted label is checked against the known label space; predictions outside this set are discarded. For regression, the forest-level ensemble aggregation (averaging across K trees) naturally dampens outlier predictions from individual leaves.

Graceful fallback. When any LLM call fails entirely (e.g., due to unparseable output or API errors), the system falls back to classical strategies: majority voting for classification or sample mean for regression at the affected node.

D Training Efficiency and Economic Cost

All LLM calls in FORESTLLM are confined to the offline training phase; test-time inference is purely tree-based with zero LLM overhead. We report training cost under a fixed configuration: $K = 5$ trees, maximum depth $D = 3$, shot $= 16$, using GPT-4o. Four datasets are selected to cover varying feature dimensionality and task types.

Complexity bound. Each tree has at most $2^D - 1$ internal nodes (split calls) and 2^D leaves (leaf calls), giving an upper bound of $K \cdot (2^{D+1} - 1) = 75$ LLM calls per forest under our default setting. Early stopping at data-sparse nodes often reduces the actual count well below this bound.

Table 13 reports the measured LLM usage during a single training run. Several observations are worth noting. (1) On standard datasets (CDC Diabetes, Insurance), training a full forest costs approximately **\$0.2–\$0.3**, with a per-tree cost around \$0.05. Even on the high-dimensional Communities dataset (103 features), the total remains under \$1.8. (2) Feature dimensionality primarily affects *token consumption* (longer feature summaries) rather than call count. Myocardial (111 features) uses only 28 calls due to early stopping from limited labeled data, yet its per-call token cost is high. (3) Wall-clock training time ranges from approximately 2 to 7 minutes across all four datasets, practical for a one-time offline procedure.

Cost scaling. The theoretical upper bound on LLM calls scales linearly with K and exponentially

Table 5: Hyperparameter search space for CART

Parameter	Search space
criterion	gini, entropy squared_error, friedman_mse, absolute_error
max_depth	3, 5, 7, 9, 11

Table 6: Hyperparameter search space for Random Forest

Parameter	Search space
n_estimators	2, 4, 8, 16, 32, 64, 128, 256
bootstrap	True, False
max_depth	3, 5, 7, 9, 11
max_features	0.5, 0.7, 1.0

Table 7: Hyperparameter search space for XGBoost

Parameter	Search space
max_depth	3, 5, 7, 9, 11
alpha	1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100
lambda	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

Table 8: Hyperparameter search space for LogReg

Parameter	Search space
C	1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100
penalty	l1, l2

Table 9: Hyperparameter search space for ElasticNet

Parameter	Search space
alpha	1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100
l1_ratio	0.0, 1.0

Table 10: Hyperparameter search space for MLP

Parameter	Search space
hidden_layer_sizes	(64,), (128,), (256,), (512,), (1024,)
alpha	0.1, 0.01, 0.001
learning_rate_init	0.1, 0.01, 0.001

Table 11: Hyperparameter search space for TabPFN

Parameter	Search space
n_estimators	2, 4, 6, 8, 10, 12

Table 12: Hyperparameter search space for FORESTLLM

Parameter	Search space
n_estimators	3, 4, 5, 6, 7, 8, 9, 10
max_depth	2, 3, 4, 5, 6, 7
bootstrap	True
max_features	0.9

Table 13: Training-time LLM usage and monetary cost per run (K=5, D=3, shot=16). Costs are estimated using GPT-4o pricing: \$2.50 / 1M input tokens and \$10.00 / 1M output tokens.

Task	Dataset	#Feat.	Calls	Tokens	Time (s)	Cost (USD)
Cls.	CDC Diabetes	21	70	82,381	176	\$0.249
Cls.	Myocardial	111	28	164,255	132	\$0.430
Cls.	Communities	103	75	662,900	426	\$1.707
Reg.	Insurance	7	74	69,826	168	\$0.232

with D (e.g., K=9, D=5 yields an upper bound of 567 calls). However, in few-shot regimes, the limited labeled data may be insufficient to support full tree growth, causing many nodes to terminate at shallow depths before reaching the maximum depth D. As a result, the actual number of calls can be substantially lower than the theoretical bound (e.g., Myocardial uses only 28 of the 75 possible calls in Table 13). Therefore, the bound should be interpreted as a worst case; in practice, the actual training cost depends not only on the forest configuration but also substantially on the inherent characteristics of the dataset.

Test-time cost. After training, all inference is performed by the learned forest with static leaf predictions. Test-time LLM token usage and cost are **exactly zero**, with inference latency comparable to standard Random Forest (millisecond-scale per sample).

E Case Study of Reasoning During Tree Splits

To examine how FORESTLLM performs semantic reasoning during decision tree construction, we visualize its generated reasoning traces, split criteria, and leaf label predictions for both classification and regression tasks (Figures 7 and 8). To ensure consistency and parseability in split decisions, we adopt structured prompting via function calling. The corresponding prompting template and function schema are shown in Figure 5, while the leaf inference prompt is provided in Figure 6.

We select two representative datasets under a 4-shot setting, one for classification and one for regression, to illustrate FORESTLLM's reasoning process:

- `Credit-g` is a binary classification dataset containing 1,000 loan applicants, with approximately 700 labeled as "good credit" and 300 as "bad credit." It consists of 20 features, including employment status, savings level, and credit history, and is widely used in credit risk prediction and interpretability research.

- `Cpu_small` is a system performance regression dataset collected from a multi-user Sun Sparcstation system. Under real-world workloads, the system records performance metrics every 5 seconds, such as exec (number of system executions per second), rchar (characters read per second), runqsz (run queue size), and freemem (available memory pages). The task is to predict the CPU's user-mode utilization (usr) based on 12 continuous system-level features.

By examining the reasoning traces generated during node splitting, we find that FORESTLLM not only identifies predictive features but also provides semantically meaningful justifications grounded in commonsense and domain-specific knowledge. For example, in the `Credit-g` dataset, the model selects attributes such as employment status, savings level, and credit history, accompanied by explanations like "stable employment is a key factor in creditworthiness" and "higher savings indicate stronger repayment ability." In the `Cpu_small` dataset, it highlights features such as rchar, runqsz, and exec, correctly associating them with system

throughput and scheduling behavior, indicating an understanding of low-level performance dynamics.

Moreover, FORESTLLM demonstrates the ability to infer semantically meaningful split thresholds by jointly leveraging the distributions of both labeled and unlabeled data. In `Cpu_small`, for instance, the model selects a `freeswap` threshold of 500,000 to distinguish between resource-constrained and resource-rich systems; it splits on `exec` at 3.0 or 4.0 to identify differences in execution efficiency; and sets `runqsz` at 500.0 or 3.0 to reflect varying levels of system load. These thresholds are not only statistically valid but also align with intuitive interpretations of performance bottlenecks, highlighting FORESTLLM's capacity for human-aligned, data-driven reasoning under limited supervision.

When assigning labels at leaf nodes, FORESTLLM eschews conventional strategies such as majority voting or numerical averaging. Instead, it performs semantic generalization over the decision path by integrating few-shot labeled examples, domain knowledge, and logical reasoning. In the `Credit-g` dataset, for example, a path characterized by"stable employment + sufficient savings + good credit history" is semantically interpreted as a low-risk credit profile, leading to the label "YES." In `Cpu_small`, a path reflecting "ample system resources and efficient execution" yields a high predicted utilization score. This path-aware reasoning framework allows FORESTLLM to transcend low-level statistical heuristics, producing label assignments that are both interpretable and contextually aligned with the underlying semantics of the task.

Overall, FORESTLLM exhibits strong semantic reasoning and interpretability throughout the decision tree construction process. Operating under few-shot conditions with limited supervision, the model effectively fuses pretrained language knowledge with distributional cues from both labeled and unlabeled data to derive meaningful split criteria and label assignments. This synergy enables FORESTLLM to overcome the inherent limitations of data scarcity, enhancing both its adaptability and the transparency of its predictions in low-resource tabular learning scenarios.

F Results with Different LLM Backbones

To evaluate the impact of different LLM backbones on the performance of FORESTLLM, we conduct additional experiments using `gpt-4-32k-0613` and `gpt-3.5-turbo-0125`, both deployed via Azure, as shown in Tables 14 and 15. Although `GPT-4-32k` is based on an older architecture and lacks the enhanced capabilities of `GPT-4o`, it delivers comparable results across a wide range of datasets and even surpasses `GPT-4o` in certain cases.

G Full Results

This section presents the complete experimental results across all datasets, baselines, and shot configurations (4, 8, 16, 32, and 48). While the main paper reports a representative subset due to space constraints, we include the full performance tables here for both classification and regression tasks, as shown in Tables 16 and 17. These tables provide a comprehensive view of model behavior across varying supervision levels and further validate the robustness of our findings.

Prompt for Node Split Condition Inference

```
## Objective
You are an expert. Given the task description, few-shot examples, and feature summary below, you are extracting
conditions as rules to solve the task.

## Task Description
{task_description}

## Few-shot Examples (Optional — include if few-shot examples exist for the current node)
{row_wise_serialized_few_shot_examples}

## Feature Summary
{feature_summary}

## Instructions
Let's first understand the problem and solve the problem step by step.
Step 1. Analyze the causal relationship or tendency between each feature and task description based on general
knowledge and common sense.
Step 2. Based on the above data distribution patterns and Step 1's results, infer one condition to split the
data into two parts, following the format below. The condition should make sense, well match features.

## Response Format
For your response,
- If splitting on a numerical feature, specify a threshold value based on the data distribution.
- If splitting on a categorical feature, specify a list of values.
```

Function to Call

```
{
    "name": "Infer_split_condition",
    "description": f"Infer a condition to split the data into two parts.",
    "parameters": {
        "properties": {
            "Reasoning": {
                "type": "string",
                "title": "reasoning",
                "description": "Detailed reasoning about the split decision based on data patterns."
            },
            "Feature": {
                "type": "string",
                "title": "Feature",
                "description": "The feature selected for the split."
            },
            "Threshold": {
                "type": ["number", "null"],
                "title": "Threshold",
                 "description": "If the selected feature to split on is numerical, specify the threshold value.
Set null if the selected feature is not numerical."
            },
            "Categorical_values": {
                "type": ["array", "null"],
                "items": {"type": "string"},
                "title": "Categorical Values",
                  "description": "If the selected feature to split on is categorical, specify the values that
should go to the left child. Set null if the selected feature is not categorical."
            }
        }
    }
}
```

Figure 5: Prompt template and function calling schema used by FORESTLLM to generate structured and interpretable split conditions. The `task_description` provides a concise natural language description of the prediction task, drawn from domain knowledge or dataset metadata (e.g., "Given this person's characteristics, does this person have diabetes?"). The `row_wise_serialized_few_shot_examples` lists labeled samples under the current node in a row-wise serialized format, serving as few-shot examples to guide the model's reasoning. The `feature_summary` summarizes the features of all samples within the current node — including both labeled and unlabeled data — describing feature types, value distributions for categorical features, and statistics for numerical features.

Prompt for Leaf Label Inference

```
## Objective
You are an expert in reasoning and classification tasks. Given the task description, decision path rule, and
few-shot examples, predict the most appropriate class label for this leaf node using general knowledge and
common sense.

## Task Description
{task_description}

## Decision Path Rule
{decision_path_rule}

## Few-shot Examples (Optional — include if few-shot examples exist for the current leaf node)
{row_wise_serialized_few_shot_examples}

## Classes (Optional — include only for classification tasks)
{classes}

## Question:
Based on the decision path rule above, which class/what value is most likely for samples that reach this leaf
node?

## Response Format
Please respond strictly in the following JSON format:

```json
{
  "reasoning": "[your brief explanation]",
  "label": "[your predicted label]"
}
```
```

Figure 6: Prompt template used by FORESTLLM for leaf node label inference.The `task_description` provides a concise natural language description of the prediction task. The `decision_path_rule` summarizes the conditions along the path from the root to the current leaf node, representing the accumulated split decisions applied to reach this node. The `row_wise_serialized_few_shot_examples` lists representative labeled samples that have arrived at the current leaf node, formatted in a row-wise serialized manner to support in-context few-shot reasoning. The `classes` enumerates all possible class labels for the classification task, including those that may not appear in the few-shot examples.

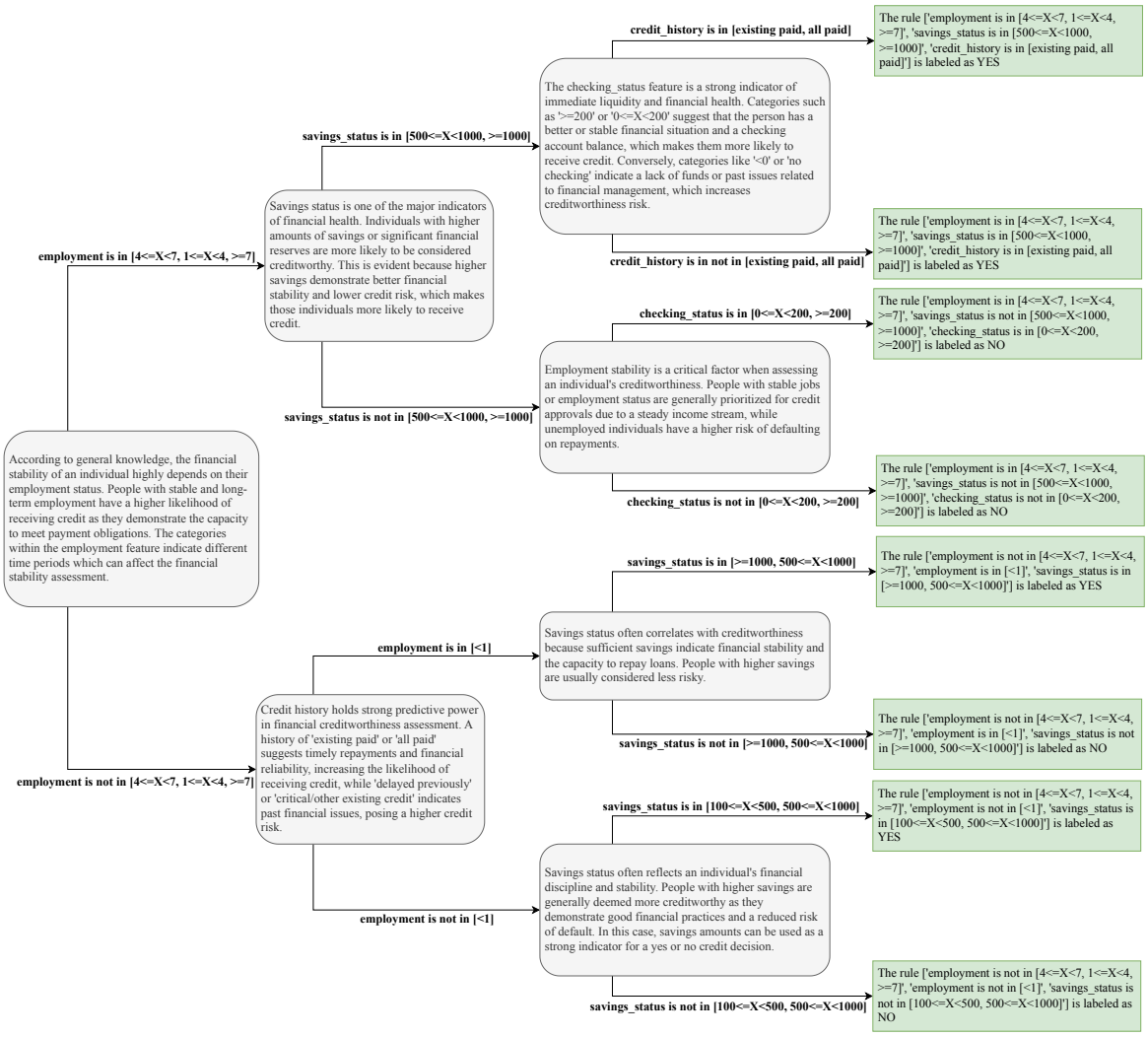

Figure 7: A decision tree generated by FORESTLLM on the Credit-g dataset.

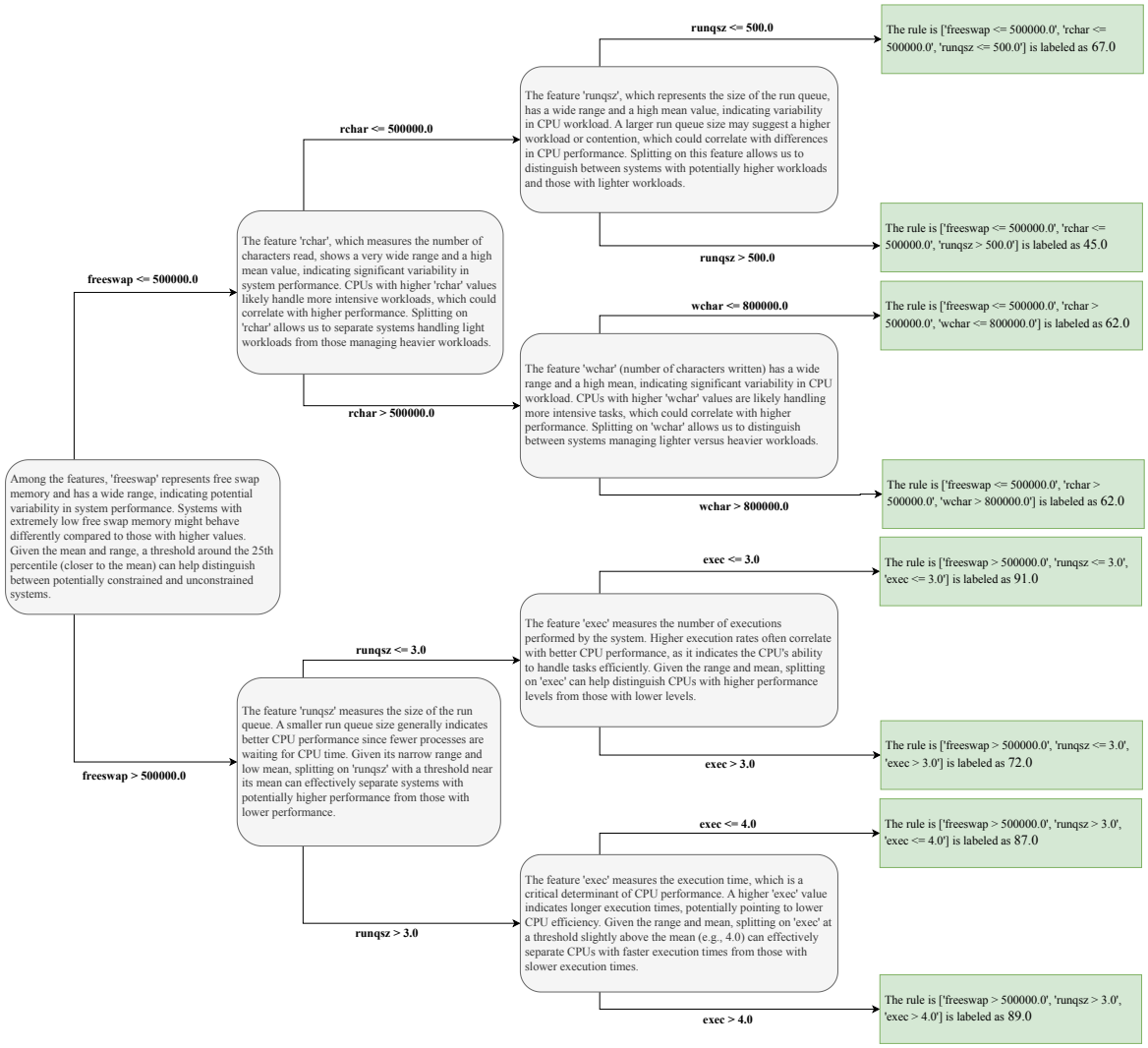

Figure 8: A decision tree generated by FORESTLLM on the Cpu_small dataset.

Table 14: Effect of LLM backbone choice on classification performance

Data	Shot	GPT-4o	GPT-4-32k	GPT-3.5-turbo
adult	4	0.833 ± 0.041	**0.838±0.030**	0.816±0.034
	8	0.827 ± 0.020	**0.839±0.021**	0.824±0.022
	16	0.823 ± 0.028	**0.835±0.027**	0.823±0.014
	32	0.846 ± 0.020	**0.847±0.020**	0.814±0.026
	48	0.834 ± 0.032	**0.852±0.018**	0.833±0.011
blood	4	0.691 ± 0.066	**0.719±0.060**	0.699±0.051
	8	0.698 ± 0.050	0.720±0.035	**0.727±0.030**
	16	0.709 ± 0.042	**0.734±0.038**	0.729±0.038
	32	0.723 ± 0.043	**0.743±0.037**	0.739±0.035
	48	0.724 ± 0.049	**0.744±0.030**	0.732±0.035
bank	4	0.837 ± 0.014	**0.843±0.009**	0.755±0.036
	8	0.835 ± 0.013	**0.842±0.012**	0.747±0.042
	16	**0.843 ± 0.008**	0.843±0.012	0.758±0.024
	32	0.841 ± 0.010	**0.847±0.009**	0.740±0.031
	48	**0.843 ± 0.011**	0.841±0.011	0.749±0.036
car	4	0.864 ± 0.034	**0.878±0.022**	0.771±0.030
	8	0.879 ± 0.028	**0.903±0.018**	0.728±0.022
	16	0.881 ± 0.021	**0.893±0.023**	0.755±0.026
	32	0.881 ± 0.029	**0.909±0.021**	0.769±0.030
	48	0.893 ± 0.020	**0.910±0.022**	0.758±0.030
communities	4	**0.784 ± 0.012**	0.774±0.006	0.643±0.043
	8	**0.782 ± 0.008**	0.771±0.008	0.627±0.037
	16	**0.789 ± 0.010**	0.772±0.010	0.642±0.042
	32	**0.781 ± 0.013**	0.780±0.010	0.660±0.026
	48	**0.779 ± 0.011**	0.774±0.011	0.661±0.046
credit-g	4	0.669 ± 0.042	**0.670±0.034**	0.625±0.019
	8	**0.687 ± 0.053**	0.686±0.050	0.628±0.030
	16	**0.693 ± 0.044**	0.679±0.039	0.650±0.034
	32	**0.709 ± 0.031**	0.679±0.034	0.630±0.034
	48	**0.713 ± 0.045**	0.704±0.035	0.627±0.030
cdc diabetes	4	0.700 ± 0.012	**0.702±0.010**	0.612±0.030
	8	0.699 ± 0.012	**0.703±0.012**	0.594±0.034
	16	**0.705 ± 0.011**	0.704±0.011	0.593±0.035
	32	0.709 ± 0.005	**0.710±0.011**	0.594±0.026
	48	0.706 ± 0.009	**0.706±0.007**	0.598±0.019
heart	4	0.887 ± 0.022	**0.894±0.015**	0.834±0.038
	8	0.896 ± 0.015	**0.899±0.012**	0.785±0.022
	16	**0.898 ± 0.011**	0.893±0.017	0.773±0.024
	32	**0.896 ± 0.013**	0.895±0.012	0.790±0.032
	48	**0.898 ± 0.018**	0.891±0.014	0.794±0.022
myocardial	4	0.653 ± 0.048	0.619±0.069	**0.667±0.037**
	8	**0.660 ± 0.046**	0.623±0.053	0.643±0.038
	16	**0.687 ± 0.048**	0.645±0.025	0.647±0.041
	32	**0.681 ± 0.055**	0.639±0.022	0.644±0.028
	48	**0.658 ± 0.034**	0.656±0.023	0.636±0.027
breast-w	4	0.992 ± 0.006	**0.993±0.005**	0.989±0.008
	8	**0.993 ± 0.006**	0.992±0.006	0.991±0.007
	16	0.993 ± 0.006	**0.994±0.005**	0.992±0.005
	32	**0.994 ± 0.004**	0.993±0.006	0.990±0.006
	48	**0.994 ± 0.006**	0.992±0.006	0.992±0.006
cultivars	4	**0.661 ± 0.031**	0.653±0.044	0.630±0.040
	8	0.653 ± 0.025	**0.657±0.032**	0.595±0.034
	16	**0.679 ± 0.027**	0.677±0.044	0.628±0.030
	32	**0.676 ± 0.041**	0.670±0.055	0.631±0.039
	48	0.670 ± 0.049	**0.671±0.045**	0.639±0.034
NHANES	4	**0.987 ± 0.010**	0.974±0.055	0.892±0.128
	8	**0.998 ± 0.002**	0.996±0.003	0.887±0.080
	16	**0.999 ± 0.001**	0.996±0.006	0.947±0.049
	32	**0.999 ± 0.001**	0.996±0.008	0.953±0.037
	48	**0.999 ± 0.001**	0.995±0.007	0.964±0.054
gallstone	4	0.658 ± 0.042	0.674±0.058	**0.698±0.037**
	8	0.653 ± 0.054	0.666±0.066	**0.679±0.045**
	16	0.662 ± 0.045	0.674±0.042	**0.677±0.058**
	32	0.659 ± 0.045	0.637±0.049	**0.667±0.033**
	48	0.663 ± 0.054	0.656±0.046	**0.689±0.038**

Table 15: Effect of LLM backbone choice on regression performance

Data	Shot	GPT-4o	GPT-4-32k	GPT-3.5-turbo
bike	4	0.156 ± 0.007	**0.155±0.009**	0.175±0.014
	8	0.149 ± 0.007	**0.145±0.006**	0.177±0.012
	16	**0.144 ± 0.011**	0.146±0.008	0.170±0.006
	32	0.142 ± 0.006	**0.142±0.004**	0.168±0.007
	48	0.141 ± 0.007	**0.139±0.004**	0.170±0.006
cpu_small	4	0.161 ± 0.038	**0.153±0.008**	0.188±0.016
	8	0.154 ± 0.024	**0.151±0.026**	0.195±0.010
	16	0.114 ± 0.027	**0.112±0.027**	0.172±0.011
	32	0.116 ± 0.022	**0.115±0.021**	0.176±0.009
	48	**0.122 ± 0.021**	0.128±0.015	0.177±0.009
diamonds	4	0.090 ± 0.010	**0.089±0.008**	0.141±0.055
	8	0.095 ± 0.013	**0.092±0.012**	0.158±0.033
	16	0.095 ± 0.007	**0.090±0.008**	0.114±0.012
	32	**0.090 ± 0.005**	0.092±0.004	0.117±0.014
	48	0.090 ± 0.005	**0.090±0.004**	0.117±0.013
forest-fires	4	0.126 ± 0.029	**0.125±0.029**	0.126±0.029
	8	0.126 ± 0.029	0.125±0.029	**0.124±0.030**
	16	0.123 ± 0.029	**0.122±0.029**	0.123±0.027
	32	**0.122 ± 0.029**	0.122±0.030	0.122±0.030
	48	**0.118 ± 0.031**	0.118±0.031	0.121±0.029
houses	4	**0.150 ± 0.030**	0.151±0.028	0.180±0.056
	8	**0.131 ± 0.011**	0.133±0.014	0.150±0.016
	16	**0.128 ± 0.011**	0.128±0.011	0.143±0.013
	32	**0.127 ± 0.011**	0.128±0.010	0.146±0.012
	48	**0.126 ± 0.011**	0.127±0.009	0.148±0.011
insurance	4	0.115 ± 0.021	**0.112±0.017**	0.141±0.029
	8	**0.107 ± 0.014**	0.107±0.015	0.133±0.023
	16	**0.105 ± 0.012**	0.105±0.014	0.150±0.018
	32	**0.100 ± 0.010**	0.100±0.012	0.153±0.022
	48	0.102 ± 0.014	**0.099±0.011**	0.157±0.025
plasma_retinol	4	0.216 ± 0.038	**0.209±0.031**	0.213±0.041
	8	0.203 ± 0.028	0.203±0.034	**0.201±0.027**
	16	0.192 ± 0.018	**0.191±0.017**	0.195±0.022
	32	**0.193 ± 0.017**	0.193±0.021	0.198±0.026
	48	**0.193 ± 0.019**	0.194±0.023	0.198±0.024
wine	4	0.141 ± 0.010	0.141±0.010	**0.134±0.004**
	8	0.137 ± 0.006	**0.135±0.006**	0.136±0.006
	16	**0.132 ± 0.002**	0.132±0.005	0.136±0.005
	32	0.131 ± 0.002	**0.130±0.003**	0.138±0.005
	48	0.131 ± 0.003	**0.130±0.003**	0.141±0.006
cultivars	4	**0.206 ± 0.021**	0.213±0.027	0.216±0.033
	8	**0.196 ± 0.017**	0.199±0.017	0.204±0.017
	16	0.198 ± 0.023	0.198±0.024	**0.198±0.019**
	32	**0.193 ± 0.019**	0.194±0.019	0.196±0.017
	48	**0.191 ± 0.022**	0.192±0.017	0.198±0.016
infrared_thermography_ temperature	4	0.108 ± 0.012	**0.105±0.012**	0.126±0.009
	8	0.102 ± 0.010	**0.100±0.010**	0.135±0.026
	16	**0.103 ± 0.008**	0.107±0.010	0.129±0.011
	32	**0.097 ± 0.007**	0.100±0.010	0.125±0.010
	48	**0.096 ± 0.009**	0.101±0.007	0.127±0.010

Table 16: Performance comparison of traditional methods and LLM-based methods across various datasets and 4-, 8-, 16-, 32-, and 48-shot settings for classification tasks. The best-performing method is bolded, and the second-best is underlined. Each value represents the AUC score, reported as the mean ± standard deviation across 10 random seeds.

Dataset	Shot	Traditional Method							LLM-based Method						
		Training Required							Fine-tuned Required		No Fine-tuning Required				
		CART	Random Forest	XGBoost	LogReg	SCARF	STUNT	TabPFN	TP-BERTa	TabLLM	TABLET	LIFT-ICL	FeatLLM	P2T	Ours
adult	4	0.600 ± 0.075	0.629 ± 0.131	0.571 ± 0.048	0.539 ± 0.057	0.595 ± 0.131	0.503 ± 0.030	0.691 ± 0.159	0.582 ± 0.096	0.800 ± 0.019	0.807 ± 0.039	0.710 ± 0.052	**0.870 ± 0.025**	0.765 ± 0.035	0.833 ± 0.041
	8	0.618 ± 0.091	0.666 ± 0.121	0.596 ± 0.081	0.603 ± 0.070	0.658 ± 0.124	0.532 ± 0.033	0.733 ± 0.132	0.610 ± 0.095	0.795 ± 0.050	0.826 ± 0.024	0.694 ± 0.039	**0.880 ± 0.009**	0.779 ± 0.036	0.827 ± 0.020
	16	0.639 ± 0.098	0.725 ± 0.092	0.685 ± 0.120	0.667 ± 0.066	0.658 ± 0.122	0.541 ± 0.025	0.746 ± 0.089	0.658 ± 0.056	0.821 ± 0.025	0.819 ± 0.020	0.667 ± 0.038	**0.877 ± 0.013**	0.777 ± 0.030	0.823 ± 0.028
	32	0.672 ± 0.060	0.759 ± 0.074	0.743 ± 0.089	0.687 ± 0.057	0.717 ± 0.097	0.532 ± 0.021	0.791 ± 0.089	0.659 ± 0.067	0.819 ± 0.029	0.820 ± 0.029	0.763 ± 0.057	**0.874 ± 0.013**	0.787 ± 0.041	0.846 ± 0.020
	48	0.672 ± 0.029	0.795 ± 0.051	0.757 ± 0.093	0.726 ± 0.076	0.763 ± 0.072	0.537 ± 0.021	0.832 ± 0.035	0.690 ± 0.024	0.832 ± 0.021	0.828 ± 0.026	0.796 ± 0.025	**0.876 ± 0.007**	0.797 ± 0.031	0.834 ± 0.032
blood	4	0.493 ± 0.098	0.500 ± 0.116	0.546 ± 0.101	0.523 ± 0.128	0.502 ± 0.138	0.562 ± 0.077	0.563 ± 0.112	0.446 ± 0.069	0.561 ± 0.130	0.626 ± 0.072	0.511 ± 0.011	0.530 ± 0.134	0.568 ± 0.113	**0.691 ± 0.066**
	8	0.542 ± 0.099	0.575 ± 0.095	0.567 ± 0.060	0.564 ± 0.127	0.617 ± 0.098	0.553 ± 0.084	0.604 ± 0.118	0.473 ± 0.055	0.578 ± 0.109	0.648 ± 0.071	0.551 ± 0.055	0.598 ± 0.121	0.618 ± 0.084	**0.698 ± 0.050**
	16	0.602 ± 0.069	0.635 ± 0.073	0.601 ± 0.087	0.673 ± 0.089	0.593 ± 0.108	0.551 ± 0.085	0.636 ± 0.080	0.497 ± 0.089	0.605 ± 0.097	0.655 ± 0.063	0.523 ± 0.052	0.621 ± 0.090	0.664 ± 0.071	**0.709 ± 0.042**
	32	0.605 ± 0.056	0.658 ± 0.065	0.682 ± 0.041	0.705 ± 0.047	0.666 ± 0.047	0.571 ± 0.067	0.692 ± 0.035	0.484 ± 0.078	0.644 ± 0.037	0.680 ± 0.032	0.618 ± 0.063	0.695 ± 0.079	0.701 ± 0.040	**0.723 ± 0.043**
	48	0.626 ± 0.064	0.683 ± 0.046	0.679 ± 0.034	0.686 ± 0.082	0.693 ± 0.052	0.571 ± 0.086	0.713 ± 0.043	0.524 ± 0.057	0.663 ± 0.058	0.699 ± 0.032	0.601 ± 0.064	0.692 ± 0.056	0.702 ± 0.033	**0.724 ± 0.049**
bank	4	0.535 ± 0.029	0.613 ± 0.050	0.514 ± 0.030	0.585 ± 0.137	0.556 ± 0.070	0.531 ± 0.103	0.602 ± 0.088	0.401 ± 0.156	0.609 ± 0.079	0.829 ± 0.016	0.616 ± 0.053	0.723 ± 0.022	0.683 ± 0.041	**0.837 ± 0.014**
	8	0.608 ± 0.089	0.699 ± 0.085	0.587 ± 0.095	0.676 ± 0.098	0.572 ± 0.075	0.544 ± 0.075	0.706 ± 0.074	0.408 ± 0.139	0.638 ± 0.058	0.830 ± 0.027	0.628 ± 0.047	0.742 ± 0.021	0.723 ± 0.047	**0.835 ± 0.013**
	16	0.642 ± 0.071	0.730 ± 0.062	0.675 ± 0.088	0.738 ± 0.060	0.584 ± 0.065	0.559 ± 0.053	0.773 ± 0.084	0.447 ± 0.123	0.661 ± 0.041	0.836 ± 0.020	0.616 ± 0.036	0.752 ± 0.025	0.740 ± 0.040	**0.843 ± 0.008**
	32	0.691 ± 0.035	0.774 ± 0.033	0.752 ± 0.034	0.717 ± 0.083	0.614 ± 0.048	0.608 ± 0.068	0.815 ± 0.040	0.530 ± 0.080	0.674 ± 0.039	0.838 ± 0.013	0.713 ± 0.029	0.768 ± 0.036	0.731 ± 0.043	**0.841 ± 0.010**
	48	0.709 ± 0.044	0.773 ± 0.027	0.779 ± 0.030	0.771 ± 0.039	0.616 ± 0.048	0.665 ± 0.037	0.829 ± 0.022	0.506 ± 0.145	0.721 ± 0.074	0.838 ± 0.008	0.727 ± 0.023	0.791 ± 0.031	0.732 ± 0.030	**0.843 ± 0.011**
car	4	0.562 ± 0.051	0.554 ± 0.051	0.557 ± 0.058	0.580 ± 0.057	0.571 ± 0.074	0.564 ± 0.026	0.612 ± 0.059	0.585 ± 0.082	0.558 ± 0.043	0.840 ± 0.057	0.799 ± 0.070	0.703 ± 0.060	0.656 ± 0.045	**0.864 ± 0.034**
	8	0.581 ± 0.061	0.643 ± 0.048	0.620 ± 0.036	0.622 ± 0.033	0.622 ± 0.059	0.609 ± 0.027	0.673 ± 0.060	0.660 ± 0.051	0.617 ± 0.041	**0.918 ± 0.017**	0.779 ± 0.054	0.702 ± 0.072	0.704 ± 0.022	0.879 ± 0.028
	16	0.669 ± 0.064	0.736 ± 0.036	0.679 ± 0.041	0.642 ± 0.082	0.673 ± 0.061	0.652 ± 0.038	0.785 ± 0.060	0.724 ± 0.025	0.628 ± 0.116	0.845 ± 0.047	0.804 ± 0.023	0.744 ± 0.086	0.760 ± 0.023	**0.881 ± 0.021**
	32	0.745 ± 0.075	0.838 ± 0.038	0.809 ± 0.038	0.707 ± 0.038	0.739 ± 0.025	0.654 ± 0.036	**0.892 ± 0.028**	0.812 ± 0.040	0.782 ± 0.050	0.858 ± 0.042	0.836 ± 0.022	0.837 ± 0.027	0.784 ± 0.022	0.881 ± 0.029
	48	0.776 ± 0.051	0.853 ± 0.035	0.861 ± 0.029	0.713 ± 0.032	0.778 ± 0.040	0.679 ± 0.054	**0.935 ± 0.021**	0.889 ± 0.037	0.854 ± 0.038	0.834 ± 0.068	0.856 ± 0.024	0.709 ± 0.131	0.805 ± 0.024	0.893 ± 0.020
communities	4	0.568 ± 0.069	0.606 ± 0.037	0.539 ± 0.069	0.559 ± 0.043	0.654 ± 0.082	0.560 ± 0.030	0.687 ± 0.052	0.469 ± 0.061	N/A	**0.799 ± 0.013**	0.686 ± 0.053	0.633 ± 0.110	0.586 ± 0.071	0.784 ± 0.012
	8	0.603 ± 0.056	0.718 ± 0.039	0.664 ± 0.060	0.586 ± 0.073	0.726 ± 0.044	0.612 ± 0.053	0.743 ± 0.053	0.519 ± 0.070	N/A	**0.798 ± 0.025**	0.725 ± 0.024	0.691 ± 0.057	0.594 ± 0.095	0.782 ± 0.008
	16	0.623 ± 0.044	0.738 ± 0.021	0.689 ± 0.072	0.639 ± 0.082	0.745 ± 0.045	0.646 ± 0.033	0.769 ± 0.040	0.556 ± 0.032	N/A	**0.794 ± 0.028**	0.725 ± 0.028	0.699 ± 0.051	0.675 ± 0.071	0.789 ± 0.010
	32	0.655 ± 0.064	0.782 ± 0.038	0.752 ± 0.038	0.727 ± 0.037	0.765 ± 0.039	0.650 ± 0.030	**0.806 ± 0.029**	0.534 ± 0.049	N/A	0.799 ± 0.023	0.668 ± 0.019	0.730 ± 0.039	0.693 ± 0.105	0.781 ± 0.013
	48	0.689 ± 0.032	0.800 ± 0.024	0.783 ± 0.027	0.742 ± 0.066	0.783 ± 0.029	0.661 ± 0.028	**0.824 ± 0.024**	0.624 ± 0.103	N/A	0.802 ± 0.016	0.637 ± 0.025	0.767 ± 0.021	0.622 ± 0.094	0.779 ± 0.011
credit-g	4	0.515 ± 0.063	0.526 ± 0.056	0.534 ± 0.079	0.521 ± 0.049	0.539 ± 0.072	0.508 ± 0.051	0.532 ± 0.059	0.467 ± 0.077	0.580 ± 0.092	0.623 ± 0.065	0.506 ± 0.018	0.498 ± 0.042	0.538 ± 0.114	**0.669 ± 0.042**
	8	0.543 ± 0.074	0.571 ± 0.094	0.546 ± 0.064	0.600 ± 0.062	0.548 ± 0.053	0.506 ± 0.059	0.584 ± 0.070	0.469 ± 0.082	0.624 ± 0.110	0.627 ± 0.063	0.499 ± 0.005	0.531 ± 0.053	0.588 ± 0.097	**0.687 ± 0.053**
	16	0.572 ± 0.068	0.599 ± 0.083	0.561 ± 0.059	0.582 ± 0.075	0.558 ± 0.058	0.527 ± 0.055	0.645 ± 0.085	0.477 ± 0.105	0.658 ± 0.098	0.632 ± 0.072	0.513 ± 0.018	0.548 ± 0.071	0.661 ± 0.059	**0.693 ± 0.044**
	32	0.595 ± 0.054	0.628 ± 0.062	0.635 ± 0.065	0.626 ± 0.053	0.642 ± 0.058	0.515 ± 0.060	0.640 ± 0.060	0.425 ± 0.142	0.685 ± 0.066	0.647 ± 0.045	0.546 ± 0.038	0.569 ± 0.082	0.682 ± 0.051	**0.709 ± 0.031**
	48	0.620 ± 0.030	0.679 ± 0.059	0.677 ± 0.046	0.669 ± 0.077	0.683 ± 0.058	0.561 ± 0.028	0.693 ± 0.043	0.491 ± 0.073	0.702 ± 0.059	0.646 ± 0.032	0.597 ± 0.048	0.616 ± 0.090	0.670 ± 0.052	**0.713 ± 0.045**
cdc diabetes	4	0.539 ± 0.041	0.526 ± 0.039	0.591 ± 0.050	0.565 ± 0.050	0.604 ± 0.061	0.557 ± 0.035	0.605 ± 0.040	0.541 ± 0.080	0.624 ± 0.013	**0.702 ± 0.015**	0.655 ± 0.008	0.604 ± 0.059	0.570 ± 0.080	0.700 ± 0.012
	8	0.551 ± 0.041	0.605 ± 0.045	0.578 ± 0.055	0.559 ± 0.051	0.622 ± 0.066	0.581 ± 0.035	0.613 ± 0.051	0.578 ± 0.069	0.638 ± 0.019	**0.703 ± 0.012**	0.656 ± 0.007	0.590 ± 0.073	0.609 ± 0.084	0.699 ± 0.012
	16	0.562 ± 0.050	0.611 ± 0.056	0.600 ± 0.069	0.590 ± 0.050	0.600 ± 0.069	0.572 ± 0.045	0.630 ± 0.070	0.602 ± 0.048	0.620 ± 0.041	0.701 ± 0.025	0.649 ± 0.016	0.633 ± 0.072	0.599 ± 0.085	**0.705 ± 0.011**
	32	0.596 ± 0.047	0.666 ± 0.038	0.657 ± 0.034	0.624 ± 0.033	0.642 ± 0.074	0.605 ± 0.025	0.691 ± 0.044	0.638 ± 0.043	0.660 ± 0.027	**0.710 ± 0.013**	0.649 ± 0.011	0.673 ± 0.045	0.685 ± 0.024	0.709 ± 0.005
	48	0.595 ± 0.031	0.672 ± 0.033	0.655 ± 0.022	0.653 ± 0.026	0.667 ± 0.018	0.610 ± 0.029	0.698 ± 0.030	0.630 ± 0.024	0.668 ± 0.021	**0.709 ± 0.008**	0.636 ± 0.013	0.684 ± 0.029	0.675 ± 0.019	0.706 ± 0.009
heart	4	0.595 ± 0.092	0.711 ± 0.123	0.580 ± 0.082	0.510 ± 0.119	0.816 ± 0.123	0.611 ± 0.062	0.767 ± 0.123	0.497 ± 0.089	0.734 ± 0.109	0.858 ± 0.017	0.652 ± 0.084	0.870 ± 0.024	0.602 ± 0.125	**0.887 ± 0.022**
	8	0.639 ± 0.104	0.721 ± 0.092	0.594 ± 0.112	0.659 ± 0.081	0.867 ± 0.045	0.594 ± 0.066	0.828 ± 0.051	0.521 ± 0.100	0.809 ± 0.066	0.858 ± 0.025	0.563 ± 0.054	0.880 ± 0.025	0.763 ± 0.073	**0.896 ± 0.015**
	16	0.695 ± 0.091	0.840 ± 0.040	0.804 ± 0.077	0.742 ± 0.108	0.891 ± 0.034	0.632 ± 0.078	0.880 ± 0.026	0.630 ± 0.090	0.848 ± 0.034	0.857 ± 0.014	0.570 ± 0.056	0.877 ± 0.037	0.763 ± 0.073	**0.898 ± 0.011**
	32	0.748 ± 0.067	0.879 ± 0.020	0.858 ± 0.017	0.794 ± 0.061	**0.900 ± 0.028**	0.643 ± 0.040	0.891 ± 0.019	0.810 ± 0.029	0.862 ± 0.047	0.854 ± 0.019	0.665 ± 0.048	0.892 ± 0.027	0.853 ± 0.043	0.896 ± 0.013
	48	0.786 ± 0.056	0.884 ± 0.020	0.877 ± 0.018	0.864 ± 0.032	**0.903 ± 0.021**	0.655 ± 0.046	0.898 ± 0.016	0.825 ± 0.026	0.871 ± 0.036	0.853 ± 0.015	0.727 ± 0.034	0.900 ± 0.017	0.854 ± 0.032	0.898 ± 0.018
myocardial	4	0.513 ± 0.038	0.519 ± 0.071	0.511 ± 0.063	0.536 ± 0.061	0.551 ± 0.079	0.527 ± 0.063	0.522 ± 0.074	0.516 ± 0.082	N/A	0.609 ± 0.073	0.503 ± 0.004	0.568 ± 0.070	0.580 ± 0.031	**0.653 ± 0.048**
	8	0.512 ± 0.047	0.537 ± 0.062	0.543 ± 0.085	0.521 ± 0.078	0.513 ± 0.078	0.528 ± 0.045	0.573 ± 0.102	0.538 ± 0.058	N/A	0.616 ± 0.047	0.519 ± 0.033	0.549 ± 0.067	0.583 ± 0.037	**0.660 ± 0.046**
	16	0.547 ± 0.060	0.592 ± 0.084	0.569 ± 0.067	0.537 ± 0.098	0.519 ± 0.048	0.538 ± 0.052	0.608 ± 0.059	0.587 ± 0.053	N/A	0.632 ± 0.033	0.570 ± 0.062	0.577 ± 0.062	0.576 ± 0.062	**0.687 ± 0.048**
	32	0.532 ± 0.072	0.559 ± 0.066	0.540 ± 0.055	0.553 ± 0.057	0.545 ± 0.091	0.531 ± 0.067	0.601 ± 0.078	0.591 ± 0.058	N/A	0.625 ± 0.040	0.572 ± 0.044	0.589 ± 0.074	0.587 ± 0.055	**0.681 ± 0.055**
	48	0.536 ± 0.061	0.561 ± 0.063	0.557 ± 0.065	0.538 ± 0.077	0.559 ± 0.100	0.515 ± 0.056	0.596 ± 0.086	0.583 ± 0.072	N/A	0.625 ± 0.033	0.580 ± 0.038	0.547 ± 0.066	0.539 ± 0.050	**0.658 ± 0.034**
breast-w	4	0.836 ± 0.066	0.960 ± 0.021	0.767 ± 0.072	0.783 ± 0.220	0.985 ± 0.009	0.883 ± 0.137	0.986 ± 0.010	0.923 ± 0.040	0.985 ± 0.010	0.974 ± 0.012	0.759 ± 0.041	0.986 ± 0.008	0.979 ± 0.013	**0.992 ± 0.006**
	8	0.856 ± 0.080	0.974 ± 0.016	0.832 ± 0.087	0.887 ± 0.114	0.983 ± 0.011	0.927 ± 0.039	0.985 ± 0.009	0.926 ± 0.036	0.983 ± 0.009	0.978 ± 0.012	0.768 ± 0.065	0.987 ± 0.008	0.981 ± 0.015	**0.993 ± 0.006**
	16	0.851 ± 0.084	0.970 ± 0.020	0.876 ± 0.078	0.962 ± 0.045	0.985 ± 0.010	0.921 ± 0.052	0.984 ± 0.009	0.934 ± 0.029	0.979 ± 0.012	0.978 ± 0.013	0.912 ± 0.051	0.988 ± 0.008	0.985 ± 0.010	**0.993 ± 0.006**
	32	0.910 ± 0.024	0.977 ± 0.014	0.949 ± 0.035	0.971 ± 0.035	0.983 ± 0.017	0.920 ± 0.043	0.988 ± 0.007	0.950 ± 0.018	0.982 ± 0.013	0.977 ± 0.012	0.960 ± 0.009	0.987 ± 0.007	0.988 ± 0.008	**0.994 ± 0.004**
	48	0.909 ± 0.048	0.966 ± 0.032	0.964 ± 0.030	0.981 ± 0.014	0.987 ± 0.010	0.929 ± 0.040	0.988 ± 0.006	0.949 ± 0.018	0.982 ± 0.009	0.978 ± 0.012	0.976 ± 0.013	0.988 ± 0.006	0.989 ± 0.006	**0.994 ± 0.006**
cultivars	4	0.514 ± 0.073	0.493 ± 0.070	0.470 ± 0.059	0.504 ± 0.064	0.494 ± 0.087	0.514 ± 0.071	0.499 ± 0.091	0.445 ± 0.085	0.524 ± 0.095	0.525 ± 0.047	0.531 ± 0.032	0.528 ± 0.050	0.536 ± 0.076	**0.661 ± 0.031**
	8	0.486 ± 0.063	0.497 ± 0.062	0.503 ± 0.062	0.528 ± 0.075	0.519 ± 0.095	0.498 ± 0.090	0.541 ± 0.069	0.495 ± 0.056	0.554 ± 0.075	0.546 ± 0.061	0.522 ± 0.066	0.561 ± 0.058	0.543 ± 0.073	**0.653 ± 0.025**
	16	0.495 ± 0.066	0.523 ± 0.076	0.510 ± 0.042	0.530 ± 0.055	0.527 ± 0.082	0.519 ± 0.093	0.511 ± 0.071	0.491 ± 0.110	0.584 ± 0.084	0.531 ± 0.092	0.563 ± 0.070	0.582 ± 0.063	0.576 ± 0.043	**0.679 ± 0.027**
	32	0.536 ± 0.083	0.525 ± 0.055	0.531 ± 0.049	0.527 ± 0.050	0.525 ± 0.071	0.540 ± 0.063	0.530 ± 0.083	0.517 ± 0.099	0.591 ± 0.062	0.527 ± 0.088	0.564 ± 0.068	0.568 ± 0.075	0.622 ± 0.057	**0.676 ± 0.041**
	48	0.547 ± 0.094	0.570 ± 0.067	0.546 ± 0.065	0.518 ± 0.052	0.519 ± 0.076	0.555 ± 0.070	0.555 ± 0.094	0.558 ± 0.095	0.608 ± 0.039	0.557 ± 0.082	0.568 ± 0.038	0.579 ± 0.094	0.615 ± 0.049	**0.670 ± 0.049**
NHANES	4	0.662 ± 0.183	0.805 ± 0.152	0.819 ± 0.177	0.870 ± 0.200	0.773 ± 0.162	0.512 ± 0.018	0.898 ± 0.098	0.457 ± 0.124	0.975 ± 0.062	0.525 ± 0.047	0.531 ± 0.032	0.527 ± 0.189	0.673 ± 0.212	**0.987 ± 0.010**
	8	0.962 ± 0.031	0.900 ± 0.137	0.880 ± 0.135	0.968 ± 0.034	0.812 ± 0.119	0.518 ± 0.016	0.976 ± 0.043	0.519 ± 0.114	**0.999 ± 0.001**	0.546 ± 0.061	0.522 ± 0.066	0.750 ± 0.116	0.925 ± 0.037	0.998 ± 0.002
	16	0.967 ± 0.023	0.974 ± 0.020	0.956 ± 0.039	0.988 ± 0.011	0.844 ± 0.116	0.532 ± 0.061	0.999 ± 0.001	0.521 ± 0.058	0.999 ± 0.002	0.531 ± 0.092	0.563 ± 0.070	0.899 ± 0.082	0.969 ± 0.018	**0.999 ± 0.001**
	32	0.988 ± 0.016	0.990 ± 0.012	0.989 ± 0.028	0.987 ± 0.011	0.835 ± 0.142	0.535 ± 0.069	**1.000 ± 0.000**	0.792 ± 0.073	1.000 ± 0.000	0.527 ± 0.088	0.564 ± 0.068	0.960 ± 0.023	0.959 ± 0.012	0.999 ± 0.001
	48	0.989 ± 0.015	0.992 ± 0.013	0.998 ± 0.007	0.990 ± 0.009	0.810 ± 0.084	0.579 ± 0.103	**1.000 ± 0.000**	0.840 ± 0.092	0.999 ± 0.002	0.557 ± 0.082	0.568 ± 0.038	0.977 ± 0.026	0.967 ± 0.008	0.999 ± 0.001
gallstone	4	0.559 ± 0.090	0.564 ± 0.085	0.481 ± 0.063	0.493 ± 0.064	0.505 ± 0.103	0.509 ± 0.053	0.570 ± 0.095	0.440 ± 0.081	0.465 ± 0.071	0.565 ± 0.071	0.533 ± 0.046	0.540 ± 0.078	0.471 ± 0.076	**0.658 ± 0.042**
	8	0.555 ± 0.094	0.571 ± 0.115	0.508 ± 0.089	0.532 ± 0.088	0.522 ± 0.097	0.505 ± 0.058	0.597 ± 0.100	0.432 ± 0.070	0.473 ± 0.102	0.558 ± 0.074	0.543 ± 0.043	0.532 ± 0.122	0.436 ± 0.067	**0.653 ± 0.054**
	16	0.592 ± 0.099	0.591 ± 0.101	0.589 ± 0.133	0.596 ± 0.102	0.537 ± 0.128	0.528 ± 0.091	0.612 ± 0.122	0.426 ± 0.071	0.461 ± 0.077	0.563 ± 0.112	0.529 ± 0.062	0.557 ± 0.083	0.481 ± 0.040	**0.662 ± 0.045**
	32	0.598 ± 0.071	0.668 ± 0.084	0.665 ± 0.086	0.667 ± 0.093	0.559 ± 0.074	0.558 ± 0.076	**0.696 ± 0.103**	0.454 ± 0.077	0.559 ± 0.053	0.631 ± 0.079	0.551 ± 0.040	0.613 ± 0.077	0.485 ± 0.074	0.659 ± 0.045
	48	0.663 ± 0.054	0.725 ± 0.076	0.740 ± 0.050	0.711 ± 0.087	0.598 ± 0.080	0.556 ± 0.092	**0.793 ± 0.052**	0.436 ± 0.098	0.548 ± 0.077	0.628 ± 0.087	0.546 ± 0.061	0.661 ± 0.053	0.499 ± 0.067	0.663 ± 0.054

Table 17: Performance comparison of traditional methods and LLM-based methods across various datasets and 4-, 8-, 16-, 32-, and 48-shot settings for regression tasks. The best-performing method is bolded, and the second-best is underlined. Each value represents the NRMSE score, reported as the mean ± standard deviation across 10 random seeds.

Dataset	Shot	Traditional Method						LLM-based Method			
		Training Required						Fine-tuned Required	No Fine-tuning Required		
		ElasticNet	MLP	CART	Random Forest	XGBoost	TabPFN	TP-BERTa	LIFT-ICL	P2T	Ours
bike	4	0.266 ± 0.080	0.225 ± 0.038	0.220 ± 0.026	0.194 ± 0.025	0.224 ± 0.055	0.192 ± 0.013	0.264 ± 0.004	0.161 ± 0.011	0.190 ± 0.020	**0.156 ± 0.007**
	8	0.211 ± 0.065	0.247 ± 0.082	0.300 ± 0.086	0.197 ± 0.024	0.243 ± 0.121	0.191 ± 0.028	0.262 ± 0.004	0.259 ± 0.058	0.185 ± 0.013	**0.149 ± 0.007**
	16	0.202 ± 0.053	0.256 ± 0.066	0.304 ± 0.132	0.200 ± 0.068	0.193 ± 0.024	0.165 ± 0.012	0.261 ± 0.004	0.156 ± 0.015	0.194 ± 0.014	**0.144 ± 0.011**
	32	0.205 ± 0.033	0.264 ± 0.106	0.200 ± 0.085	0.170 ± 0.077	0.184 ± 0.084	**0.140 ± 0.053**	0.259 ± 0.004	0.152 ± 0.015	0.165 ± 0.060	0.142 ± 0.006
	48	0.184 ± 0.016	0.208 ± 0.076	0.215 ± 0.101	0.153 ± 0.059	0.153 ± 0.055	**0.126 ± 0.047**	0.258 ± 0.004	0.155 ± 0.020	0.176 ± 0.009	0.141 ± 0.007
cpu_small	4	0.322 ± 0.157	49.042 ± 82.905	0.168 ± 0.050	0.180 ± 0.017	0.198 ± 0.019	0.167 ± 0.024	0.789 ± 0.003	0.173 ± 0.082	0.234 ± 0.071	**0.161 ± 0.038**
	8	0.389 ± 0.314	9.865 ± 27.918	0.172 ± 0.019	0.167 ± 0.022	0.187 ± 0.009	0.164 ± 0.029	0.770 ± 0.004	0.172 ± 0.036	0.167 ± 0.022	**0.154 ± 0.024**
	16	0.335 ± 0.437	0.987 ± 0.375	0.156 ± 0.067	0.122 ± 0.040	0.180 ± 0.037	0.118 ± 0.039	0.745 ± 0.004	**0.101 ± 0.045**	0.147 ± 0.036	0.114 ± 0.027
	32	0.234 ± 0.146	1.674 ± 2.491	0.101 ± 0.047	0.098 ± 0.048	0.151 ± 0.057	0.114 ± 0.060	0.720 ± 0.004	**0.098 ± 0.048**	0.101 ± 0.043	0.116 ± 0.022
	48	0.230 ± 0.142	0.867 ± 0.003	**0.084 ± 0.051**	0.094 ± 0.047	0.129 ± 0.053	0.129 ± 0.056	0.704 ± 0.004	0.090 ± 0.052	0.090 ± 0.040	0.122 ± 0.021
diamonds	4	0.142 ± 0.063	0.153 ± 0.038	0.152 ± 0.020	0.152 ± 0.021	0.156 ± 0.019	0.157 ± 0.078	0.302 ± 0.002	**0.080 ± 0.009**	0.116 ± 0.021	0.090 ± 0.010
	8	0.169 ± 0.053	0.116 ± 0.026	0.150 ± 0.057	0.118 ± 0.019	0.129 ± 0.014	0.153 ± 0.103	0.302 ± 0.002	**0.080 ± 0.007**	0.109 ± 0.014	0.095 ± 0.013
	16	0.123 ± 0.031	0.115 ± 0.028	0.126 ± 0.019	0.120 ± 0.015	0.134 ± 0.021	0.119 ± 0.059	0.302 ± 0.002	**0.082 ± 0.011**	0.109 ± 0.008	0.095 ± 0.007
	32	0.092 ± 0.013	0.101 ± 0.029	0.116 ± 0.017	0.096 ± 0.009	0.107 ± 0.009	0.089 ± 0.020	0.302 ± 0.002	**0.084 ± 0.013**	0.095 ± 0.006	0.090 ± 0.005
	48	0.087 ± 0.009	0.094 ± 0.037	0.112 ± 0.017	0.095 ± 0.010	0.097 ± 0.010	0.089 ± 0.035	0.302 ± 0.002	**0.078 ± 0.011**	0.092 ± 0.006	0.090 ± 0.005
forest-fires	4	0.128 ± 0.031	0.130 ± 0.031	0.127 ± 0.029	0.127 ± 0.030	0.127 ± 0.030	0.127 ± 0.029	0.128 ± 0.030	0.127 ± 0.030	0.127 ± 0.030	**0.126 ± 0.029**
	8	0.170 ± 0.109	0.128 ± 0.030	0.204 ± 0.221	0.143 ± 0.042	0.140 ± 0.036	0.170 ± 0.131	0.127 ± 0.029	0.153 ± 0.071	0.136 ± 0.033	**0.126 ± 0.029**
	16	0.147 ± 0.051	0.133 ± 0.034	0.214 ± 0.192	0.146 ± 0.046	0.144 ± 0.044	0.136 ± 0.037	0.126 ± 0.029	0.158 ± 0.070	0.143 ± 0.045	**0.123 ± 0.029**
	32	0.127 ± 0.027	0.132 ± 0.029	0.162 ± 0.079	0.139 ± 0.037	0.128 ± 0.030	0.125 ± 0.029	0.126 ± 0.030	0.149 ± 0.053	**0.121 ± 0.054**	0.122 ± 0.029
	48	0.126 ± 0.027	0.152 ± 0.072	0.197 ± 0.112	0.136 ± 0.037	0.136 ± 0.031	0.125 ± 0.028	0.126 ± 0.029	0.160 ± 0.052	0.119 ± 0.052	**0.118 ± 0.031**
houses	4	0.393 ± 0.232	6.056 ± 2.924	0.252 ± 0.044	0.180 ± 0.034	0.156 ± 0.020	0.176 ± 0.023	1.270 ± 0.143	0.214 ± 0.085	0.358 ± 0.209	**0.150 ± 0.030**
	8	0.286 ± 0.211	6.218 ± 6.041	0.181 ± 0.028	0.170 ± 0.015	0.161 ± 0.023	0.142 ± 0.023	0.801 ± 0.108	0.169 ± 0.032	0.192 ± 0.052	**0.131 ± 0.011**
	16	0.183 ± 0.037	2.840 ± 1.398	0.175 ± 0.025	0.150 ± 0.012	0.156 ± 0.015	0.131 ± 0.012	0.285 ± 0.059	0.148 ± 0.013	0.163 ± 0.018	**0.128 ± 0.011**
	32	0.134 ± 0.024	3.275 ± 4.016	0.174 ± 0.025	0.135 ± 0.017	0.142 ± 0.020	**0.112 ± 0.013**	0.172 ± 0.014	0.153 ± 0.021	0.149 ± 0.014	0.127 ± 0.011
	48	0.121 ± 0.018	2.081 ± 2.099	0.162 ± 0.022	0.131 ± 0.012	0.131 ± 0.016	**0.104 ± 0.012**	0.172 ± 0.014	0.156 ± 0.025	0.141 ± 0.011	0.126 ± 0.011
insurance	4	0.240 ± 0.044	0.235 ± 0.056	0.282 ± 0.070	0.236 ± 0.035	0.236 ± 0.027	0.194 ± 0.041	0.325 ± 0.039	0.125 ± 0.018	0.186 ± 0.055	**0.115 ± 0.021**
	8	0.224 ± 0.045	0.207 ± 0.023	0.295 ± 0.126	0.255 ± 0.065	0.245 ± 0.042	0.163 ± 0.038	0.325 ± 0.039	0.116 ± 0.020	0.145 ± 0.026	**0.107 ± 0.014**
	16	0.185 ± 0.017	0.193 ± 0.012	0.230 ± 0.049	0.232 ± 0.028	0.234 ± 0.031	0.158 ± 0.048	0.325 ± 0.039	0.116 ± 0.017	0.119 ± 0.044	**0.105 ± 0.012**
	32	0.156 ± 0.023	0.158 ± 0.021	0.181 ± 0.086	0.212 ± 0.079	0.207 ± 0.077	0.117 ± 0.045	0.325 ± 0.039	0.112 ± 0.015	0.119 ± 0.013	**0.100 ± 0.010**
	48	0.153 ± 0.020	0.162 ± 0.030	0.222 ± 0.056	0.236 ± 0.030	0.233 ± 0.037	0.122 ± 0.017	0.325 ± 0.039	0.112 ± 0.016	0.114 ± 0.042	**0.102 ± 0.014**
plasma_retinol	4	0.490 ± 0.329	0.396 ± 0.112	0.292 ± 0.100	0.239 ± 0.049	**0.215 ± 0.028**	0.227 ± 0.046	0.633 ± 0.140	0.331 ± 0.112	0.412 ± 0.163	0.216 ± 0.038
	8	0.388 ± 0.101	0.472 ± 0.248	0.304 ± 0.097	0.240 ± 0.063	0.234 ± 0.048	0.242 ± 0.060	0.631 ± 0.140	0.340 ± 0.108	0.316 ± 0.076	**0.203 ± 0.028**
	16	0.301 ± 0.171	0.445 ± 0.371	0.296 ± 0.080	0.217 ± 0.038	0.220 ± 0.025	0.203 ± 0.023	0.628 ± 0.139	0.340 ± 0.129	0.268 ± 0.064	**0.192 ± 0.018**
	32	0.315 ± 0.281	0.439 ± 0.410	0.307 ± 0.084	0.231 ± 0.040	0.215 ± 0.032	0.203 ± 0.024	0.626 ± 0.138	0.327 ± 0.079	0.270 ± 0.053	**0.193 ± 0.017**
	48	0.216 ± 0.031	0.389 ± 0.268	0.287 ± 0.083	0.214 ± 0.025	0.210 ± 0.029	0.202 ± 0.025	0.624 ± 0.138	0.309 ± 0.051	0.268 ± 0.053	**0.193 ± 0.019**
wine	4	0.206 ± 0.065	0.361 ± 0.185	0.209 ± 0.030	0.170 ± 0.027	0.169 ± 0.030	0.166 ± 0.029	0.150 ± 0.005	0.164 ± 0.023	0.252 ± 0.068	**0.141 ± 0.010**
	8	0.193 ± 0.042	0.228 ± 0.077	0.188 ± 0.033	0.167 ± 0.032	0.161 ± 0.024	0.153 ± 0.023	0.146 ± 0.003	0.158 ± 0.010	0.183 ± 0.026	**0.137 ± 0.006**
	16	0.178 ± 0.032	0.207 ± 0.060	0.187 ± 0.021	0.156 ± 0.018	0.158 ± 0.016	0.155 ± 0.023	0.148 ± 0.004	0.158 ± 0.011	0.161 ± 0.018	**0.132 ± 0.002**
	32	0.157 ± 0.022	0.170 ± 0.024	0.186 ± 0.009	0.146 ± 0.007	0.149 ± 0.009	0.137 ± 0.006	0.154 ± 0.012	0.158 ± 0.011	0.135 ± 0.047	**0.131 ± 0.002**
	48	0.143 ± 0.010	0.161 ± 0.016	0.178 ± 0.017	0.141 ± 0.006	0.141 ± 0.006	0.135 ± 0.006	0.151 ± 0.009	0.159 ± 0.007	0.133 ± 0.047	**0.131 ± 0.003**
cultivars	4	0.325 ± 0.143	0.314 ± 0.056	0.268 ± 0.056	0.234 ± 0.024	0.218 ± 0.023	0.300 ± 0.240	1.403 ± 0.151	0.370 ± 0.131	0.328 ± 0.103	**0.206 ± 0.021**
	8	0.369 ± 0.221	0.330 ± 0.097	0.263 ± 0.039	0.235 ± 0.049	0.224 ± 0.039	0.221 ± 0.040	1.402 ± 0.151	0.321 ± 0.121	0.288 ± 0.085	**0.196 ± 0.017**
	16	0.224 ± 0.034	0.369 ± 0.284	0.277 ± 0.042	0.234 ± 0.035	0.231 ± 0.041	0.217 ± 0.032	1.400 ± 0.151	0.270 ± 0.074	0.266 ± 0.077	**0.198 ± 0.023**
	32	0.214 ± 0.025	0.248 ± 0.043	0.273 ± 0.041	0.216 ± 0.021	0.223 ± 0.025	0.211 ± 0.024	1.398 ± 0.151	0.266 ± 0.065	0.259 ± 0.086	**0.193 ± 0.019**
	48	0.209 ± 0.023	0.227 ± 0.029	0.260 ± 0.039	0.212 ± 0.025	0.212 ± 0.021	0.206 ± 0.023	1.396 ± 0.151	0.267 ± 0.062	0.272 ± 0.099	**0.191 ± 0.022**
infrared_thermography_temperature	4	0.152 ± 0.025	0.288 ± 0.148	0.130 ± 0.025	0.116 ± 0.018	0.151 ± 0.021	0.138 ± 0.057	6.628 ± 0.707	0.129 ± 0.017	1.637 ± 1.137	**0.108 ± 0.012**
	8	0.132 ± 0.042	0.138 ± 0.049	0.129 ± 0.016	0.109 ± 0.022	0.160 ± 0.049	0.110 ± 0.024	5.898 ± 0.628	0.105 ± 0.013	0.722 ± 1.062	**0.102 ± 0.010**
	16	0.120 ± 0.048	0.140 ± 0.049	0.117 ± 0.017	0.099 ± 0.012	0.109 ± 0.016	0.112 ± 0.058	5.036 ± 0.536	**0.098 ± 0.011**	0.106 ± 0.014	0.103 ± 0.008
	32	0.109 ± 0.085	0.121 ± 0.037	0.103 ± 0.012	**0.085 ± 0.012**	0.093 ± 0.009	0.093 ± 0.046	3.516 ± 0.375	0.095 ± 0.014	0.096 ± 0.011	0.097 ± 0.007
	48	0.079 ± 0.006	0.437 ± 0.619	0.103 ± 0.015	0.084 ± 0.010	0.088 ± 0.009	**0.078 ± 0.015**	1.842 ± 0.678	0.096 ± 0.012	0.140 ± 0.152	0.096 ± 0.009