Preprocessing Robustness in Heterogeneous Tabular Learning: Sensitivity and Volatility Indices Across Model Families

Anonymous authors
Paper under double-blind review

Abstract

Heterogeneous tabular pipelines combine numerical, categorical, and textual fields, making upstream preprocessing a major, yet often hidden, source of performance variation. While tabular architectures are heavily benchmarked, their robustness to preprocessing choices is rarely studied systematically. We address this gap by treating preprocessing as a first-class experimental variable. By defining a modular space of common transformations, including scaling, encoding, dimensionality reduction, and feature selection, we run controlled, single-knob ablations across three model families: gradient-boosted trees, MLP-style tabular networks, and tabular foundation models (TFMs). To quantify this impact, we introduce two model-agnostic robustness measures: the Preprocessing Sensitivity Index (Sens), which captures the effect of individual operations, and the Model Volatility Index (Vol), which aggregates sensitivity to measure overall fragility. To distinguish avoidable user errors from within-family variability, we evaluate both mismatched and "best-practice" operator spaces on text-rich public datasets. Our results reveal that preprocessing effects are family-dependent. Tree-based models exhibit the highest volatility; MLPs are more stable overall but remain highly vulnerable to specific representation shifts; and despite their promises of automation, TFMs do not uniformly abstract away preprocessing, showing notable sensitivity to text and categorical handling. Finally, we introduce the Preprocessing Robustness Evaluation Framework (PREF), an open-source toolkit that generates Robustness Cards to summarize a model's volatility, critical failure risks, and high-payoff tuning opportunities.

1 Introduction

Tabular prediction pipelines rarely consist of a learner alone. In real deployments, a model consumes a *representation* produced by upstream preprocessing, including numerical scaling and shaping, categorical and text encoding, feature selection, and dimensionality reduction. While modern models increasingly embed default preprocessing for numerical and categorical features, these generic transformations remain largely agnostic to the semantic nature of the data. In heterogeneous tables, mixing numerical, categorical, and text fields, such preprocessing choices—particularly when informed by feature semantics—remain a primary source of performance variation rather than mere implementation details. As a result, comparisons between tabular methods often conflate architectural differences with representation choices, and it remains unclear *which preprocessing decisions matter, for which model families, and how strongly.*

This question has become more acute with the emergence of *tabular foundation models* (TFMs), such as TabPFN, TabICL, or CTBM[1] (Spinaci et al., 2025; Hollmann et al., 2023; Qu et al., 2025), which promise "out-of-the-box" performance by amortizing inductive bias through pretraining or meta-learning. Unlike classical pipelines, many TFMs include an *internal* preprocessing stack that maps user tables into the representation used during pretraining (e.g., feature-type handling, scaling, or ensembling-time randomization). Consequently, practitioner-style upstream preprocessing composes with (rather than replaces) the model's internal interface, and seemingly reasonable transformations can induce compatibility shifts relative to the pretrained operating point. A central practical question is therefore: *Do TFMs actually reduce preprocessing fragility compared to boosted trees or modern tabular MLPs, or do they introduce new failure modes?*

[1]CTBM stands for *Commercial Tabular Foundation Model*, a proprietary model (details redacted for double-blind review).

We study preprocessing as a first-class experimental variable. We define a modular *knob space* of swappable preprocessing operations, including numerical scaling, categorical and text encoding, dimensionality reduction, feature selection, and feature augmentation, and run controlled *single-knob ablations* in which exactly one operation is changed while the remainder of the pipeline is held fixed. We perform this analysis across three representative model families: (i) gradient-boosted decision trees, namely XGBoost, LightGBM, and CatBoost (Chen & Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018), (ii) MLP-style tabular networks (Gorishniy et al., 2021; Borisov et al., 2024; Holzmüller et al., 2025; Gorishniy et al., 2025), and (iii) inference-only TFMs. To separate avoidable user mistakes from within-family variability, we evaluate two operator spaces: (i) an out-of-range stress-test space capturing common-but-mismatched choices, and (ii) an in-range space restricted to practices curated as reasonable for each family.

To summarize preprocessing robustness beyond dataset-specific anecdotes, we introduce two simple, model-agnostic indices computed from performance deltas under single-knob perturbations. The *Preprocessing Sensitivity Index* (Sens) measures how strongly a given knob typically affects a model family across datasets, while the *Model Volatility Index* (Vol) aggregates sensitivity across knobs to quantify overall preprocessing fragility. Both indices are summaries (means of absolute deltas), and we report them under both raw metrics and a dataset-normalized variant to facilitate cross-dataset aggregation.

Using seven public, text-rich tabular datasets, we find that preprocessing sensitivity is strongly family-dependent. Tree ensembles exhibit the highest aggregate volatility due to their sensitivity to categorical/text representations and feature space expansion, despite being comparatively insensitive to numerical scaling. Although rescaling continuous variables has little effect on trees, they can be sensitive to increased dimensionality and redundant features. In particular, PLE-style transformations, designed to improve neural network performance by decomposing variables into simpler components, mechanically expand the feature space and fragment non-linear patterns that trees would otherwise capture naturally. Conversely, MLP-style models exhibit lower overall volatility, but possess critical vulnerabilities to specific representation shifts, with scaling and reduction choices acting as critical knobs. Answering our central question, we find that TFMs do not abstract away the need for preprocessing: some models show pronounced sensitivity to upstream text and categorical handling, consistent with compatibility effects between external preprocessing and the model's internal input interface. We further show that, while restricting preprocessing to in-range best practices reduces volatility for classical models, the fragility of TFMs remains substantial.

Finally, we package these measurements into the *Preprocessing Robustness Evaluation Framework* (**PREF**), which takes as input datasets, a model family, and a preprocessing knob space and outputs *Robustness Cards* that summarize volatility, critical knobs (risks), and expected-payoff knobs (tuning opportunities). These cards complement accuracy-centric leaderboards by making safe operating ranges explicit.

Contributions. (i) We propose PREF: a modular evaluation framework for preprocessing robustness in heterogeneous tabular learning. (ii) We introduce Sens and Vol as model-agnostic indices for knob-wise sensitivity and overall volatility. (iii) We provide a controlled empirical study across boosted trees, tabular MLPs, and TFMs on text-rich datasets, highlighting failure modes and robustness gaps. (iv) We will release an open-source toolkit that generates Robustness Cards to support practitioner-facing model selection.

2 Related Work

We position our work at the intersection of tabular benchmarking, AutoML, and robustness analysis. Recent benchmark suites, as reported in Table 1, focus on architecture search, hyperparameter optimization (HPO), or large-scale reproducibility. For example, Klein & Hutter (2019) introduce a suite of tabular benchmarks designed for joint architecture and hyperparameter optimization. Far less effort has been put in systematic evaluation of data preprocessing, as either a fixed standardized pipeline or a component embedded in AutoML search. As a result, it is hard to quantify the impact of preprocessing on the overall model performance.

Indeed, different benchmarks assess solutions on academic or industrial datasets (Rubachev et al., 2024; Erickson et al., 2025; Kim et al., 2024; Lee et al., 2025; Liu et al., 2024), but they rarely focus on data preprocessing impact. Most rely on fixed pipelines or treat preprocessing as an opaque detail of the "AutoML" search (e.g., auto-sklearn (Feurer et al., 2015)), focusing on argmax performance, while we study preprocessing-induced

Table 1: Literature comparison and positioning. While prior benchmarks fix preprocessing or treat it as opaque feature engineering, we treat it as the primary variable of interest via single-knob ablations.

Paper	Datasets	Modalities			Model families			Prepro. protocol	Eval. focus	Outputs
		Num	Cat	Text	Trees	Deep	TFM			
TabArena Erickson et al. (2025)	51 public	✓	✓		✓	✓	✓	Fixed preprocessing	Performance model tuning + ensembling	Public leaderboard
TabReD Rubachev et al. (2024)	8 private	✓	✓		✓	✓		Fixed preprocessing Engineered features	Stress-tests temporal drift, engineered schemas	Dataset suite
TabZilla McElfresh et al. (2023)	36 public	✓	✓		✓	✓		Fixed preprocessing	Large-scale, HPO	Benchmark suite
MultiTab Lee et al. (2025)	196 public	✓	✓		✓	✓		Fixed preprocessing	Multi-dimensional reporting	Benchmark suite, regime taxonomy
Data-Centric Tabular Evaluation Tschalzev et al. (2024)	10 public	✓	✓		✓	✓		Fixed preprocessing Expert-level pipelines	Feature engineering and HPO	Benchmark framework, curated pipelines
Benchmarking TFMs with Text Mráz et al. (2025)	13 public	✓	✓	✓	✓		✓	Varying text preprocessing	Mixed-modality evaluation	Dataset suite, text pipelines
Categorical Data Encoding for AI Bolikulov et al. (2024)	1 public	✓	✓					17 cat. encoders Other steps fixed	Encoder effect on classification metrics	Pipeline description
PREF This work	7 public	✓	✓	✓	✓	✓	✓	Systematic single-knob ablations, two operator spaces	Preprocessing robustness	Framework, Toolkit for Robustness Cards

variance. Our contribution is a modular, *swap-one-knob-at-a-time* study across model families and datasets, moving beyond leaderboard rankings to provide practical guidelines on model volatility.

Existing work in data-centric evaluation (Tschalzev et al., 2024) shows that expert-engineered preprocessing improves leaderboard rankings dramatically. Yet these studies compare entire pipeline configurations rather than decomposing preprocessing into modular operations. Consequently, they provide evidence of pipeline sensitivity but do not offer a knob-wise quantification of preprocessing influence across model families. A closely related discussion argued that tree-based models retain advantages on tabular benchmarks and highlighted the role of inductive biases and data representation (Grinsztajn et al., 2022), but did not systematically benchmark preprocessing as an experimental factor across model families.

More studies investigate individual preprocessing components, such as categorical encoders (Bolikulov et al., 2024) and discretization (Rabanser et al., 2020), but typically restrict analysis to a single dataset or model class. Mixed-modality benchmarks that incorporate text columns (Mráz et al., 2025) vary integration strategies but fix one encoding per model and do not consider sensitivity metrics. While TFMs are increasingly evaluated on these mixed-modality tables, their interactions with upstream preprocessing remain unexplored.

Stability and Robustness. Preprocessing can be viewed as a structured perturbation of the learning pipeline. Classical *algorithmic stability* studies how the output of a learning algorithm changes under small perturbations of the training set (e.g., removing or modifying a sample), and relates such sensitivity to generalization (Bousquet & Elisseeff, 2002). In a complementary vein, *global sensitivity analysis* quantifies how variability in an output of interest can be attributed to variations in input factors (Saltelli, 2008). Most existing stability and robustness notions focus on perturbations at the sample level or on distribution shift, whereas our focus is on *representation-level* perturbations induced by feature construction.

Our evaluation treats the preprocessing pipeline as a parameterized mapping from raw tables to model inputs, and measures the induced change in predictive performance under controlled *single-knob* interventions. The proposed Preprocessing Sensitivity Index and Model Volatility Index can be interpreted as robust, finite-difference summaries of performance sensitivity along this structured axis of perturbations, enabling cross-family comparisons of how brittle (or adaptable) models are to upstream representation changes.

Positioning. Our contribution is not a new architecture or a larger benchmark suite, but a modular experimental framework that quantifies preprocessing-induced performance variability across trees, neural networks, and tabular foundation models. We move beyond leaderboard-centric evaluation and introduce robustness-oriented metrics that characterize model volatility w.r.t. preprocessing choices.

3 Quantifying Preprocessing Impact

3.1 Notation

Let $\mathcal{D}$ be a collection of tabular datasets and $\mathcal{M}$ a set of model families (gradient-boosted decision trees, MLP-style tabular networks, foundation models for tabular data). We denote by $d \in \mathcal{D}$ a dataset and by $m \in \mathcal{M}$ a model family. Each family $m \in \mathcal{M}$ is associated with a set of models $\mathcal{G}_m$ (e.g., XGBoost), and we denote by $g \in \mathcal{G}_m$ a specific model. We first define our performance measures and deltas for a specific model $g \in \mathcal{G}_m$, and later aggregate them to report family-level summaries for m.

We represent data preprocessing as a set of *knobs* (or operation types), indexed by $k \in \mathcal{K}$. Each knob corresponds to a *model-agnostic* operation applied at the dataset/pipeline level, such as numeric feature scaling, categorical and text encoding, dimensionality reduction, feature selection and augmentation.

For each knob k, we denote its set of admissible choices by $\mathcal{A}_k$. A preprocessing pipeline P is specified by selecting one alternative $a_k \in \mathcal{A}_k$ per knob $k \in \mathcal{K}$. The knobs in $\mathcal{K}$ exclude *model-specific transformations*, such as internal feature normalization in neural networks; these model-specific operations are considered part of the model family m, while $\mathcal{K}$ only contains dataset-level, model-agnostic preprocessing.

When we change a preprocessing knob, we reset the model and re-execute the standard learning or inference procedure adhering to the original configuration.

Performance metrics. Given a specific model $g \in \mathcal{G}_m$, dataset d, and preprocessing pipeline P, we write

$$F_{\text{raw}}(g, d, P) \in \mathbb{R}$$

for the resulting predictive performance under the *raw task metric*. For example, F_{raw} can be the mean cross-validated accuracy estimated across all (g, d) pairs using a fixed split protocol and multiple random seeds. We also consider a dataset-normalized version. For each dataset d, let

$$F_d^{\max} \;=\; \max_{g,P} F_{\text{raw}}(g, d, P), \qquad F_d^{\min} \;=\; \min_{g,P} F_{\text{raw}}(g, d, P),$$

taken over all model-pipeline combinations evaluated on d. We then define

$$F_{\text{norm}}(g, d, P) \;=\; \frac{F_{\text{raw}}(g, d, P) - F_d^{\min}}{F_d^{\max} - F_d^{\min} + \varepsilon}, \tag{1}$$

where $\varepsilon > 0$ is a small constant to avoid division by zero. Thus $F_{\text{norm}}(g, d, P) \in [0, 1]$ expresses performance as a relative position between the worst and best observed configurations on d. We will write F generically and instantiate it either as F_{raw} or F_{norm}; both yield valid sensitivity indices. The raw variant preserves the scale of the original metric, while the normalized variant facilitates cross-dataset aggregation.

3.2 Operator space of analysis.

Although $\mathcal{A}_k$ denotes the full set of implemented alternatives for knob k, we distinguish two *operator spaces* that correspond to two user-facing robustness questions. For each model family m and knob k, we define:

- **Wrong-choice penalties** (WC): a subset $\mathcal{A}_k^{\text{WC}}(m) \subseteq \mathcal{A}_k$ comprising common preprocessing choices, including those that are mismatched or inappropriate for family m. This measures: *How badly can things go if users apply a typical yet mismatched preprocessing choice?*
- **Best-practice robustness** (BP): a subset $\mathcal{A}_k^{\text{BP}}(m) \subseteq \mathcal{A}_k$ capturing practices curated as reasonable for family m. This measures: *How much does performance change when we vary preprocessing within the recommended operating range of the family?*

By defining $\mathcal{A}_k^{\mathrm{BP}}(m) \subseteq \mathcal{A}_k^{\mathrm{WC}}(m)$, we isolate the variability of a model's performance within its recommended operating range, removing out-of-range choices. Choices for Best-Practice alternatives were made according to commonly used and documented practices for data preprocessing. We report the knobs and their methods in Table 2, and provide the specific method selections per knob and model family in Appendix A.

3.3 Reference Pipelines and Single-Knob Ablations

Our indices are based on comparing each pipeline to a *reference* (or baseline) pipeline via "one-knob-at-a-time" ablations. We consider three complementary reference regimes.

Classic. A pipeline that requires minimal dataset-specific human effort and mirrors common, low-friction setups in the tabular literature (e.g., a basic categorical/text handling and no manual feature engineering). This regime is attractive for low setup cost and ease of automation.
Model-wise Classic. A pipeline reflecting common practitioner usage per model family. It uses a shared coarse design across datasets, while adapting choices known to be compatible with each family. This reference remains lightweight (no dataset-specific feature engineering) and reduces avoidable mismatches.
Model-wise Best. A pipeline that selects, for each model family m on dataset d, the best-performing preprocessing pipeline among the configurations evaluated. This reference highlights the achievable performance envelope of each model and quantifies the potential gains from preprocessing optimization.

In the following we write $P_{g,d}^{\mathrm{ref}}$ for a generic reference pipeline. Sensitivity indices can be computed in each regime separately, allowing us to contrast how fragile a model is with a simple versus a stronger pipeline.

Single-knob ablations. We consider alternative pipelines that differ from $P_{g,d}^{\mathrm{ref}}$ in exactly one knob. Specifically, for a given knob $k \in \mathcal{K}$ and alternative $a \in \mathcal{A}_k$ distinct from the reference choice, we define the pipeline $P_{g,d}^{(k=a)}$ to be identical to $P_{g,d}^{\mathrm{ref}}$ except that knob k is set to a. For each such pipeline, we retrain the model g on dataset d using the fixed training procedure and hyperparameters for (g, d).

For a model g belonging to family m, we compute *performance deltas* under an operator space $s \in \{\mathrm{WC}, \mathrm{BP}\}$ by restricting alternatives to $a \in \mathcal{A}_k^s(m)$:

$$\Delta_{g,d,k,a}^s \;=\; F\big(g, d, P_{g,d}^{(k=a)}\big) \;-\; F\big(g, d, P_{g,d}^{\mathrm{ref}}\big). \tag{2}$$

When $F = F_{\mathrm{raw}}$ and the raw metric is "higher is better", positive values of $\Delta_{g,d,k,a}^s$ indicate an improvement over the reference pipeline. When $F = F_{\mathrm{norm}}$, $\Delta_{g,d,k,a}^s$ measures the *relative* change in dataset-normalized performance. In practice, F is estimated over multiple random seeds, so that the deltas in equation 2 primarily reflect preprocessing effects rather than run-to-run variation.

3.4 Aggregated Measure over the Knob

The quantity $\Delta_{g,d,k,a}^s$ in equation 2 measures the effect of one alternative a for knob k, relative to the reference choice. For our indices, we first aggregate over alternatives $a \in \mathcal{A}_k^s(m)$.

A simple and robust summary of the magnitude of the effect of knob k on (g, d) is given by the mean absolute delta across alternatives. For a fixed operator space $s \in \{\mathrm{WC}, \mathrm{BP}\}$, we aggregate over alternatives $a \in \mathcal{A}_k^s(m)$ via the mean absolute delta:

$$\Delta_{g,d,k}^s \;=\; \mathrm{mean}_{a \in \mathcal{A}_k^s(m)} \big|\Delta_{g,d,k,a}^s\big|. \tag{3}$$

This aggregates over alternative settings for knob k, while being stable to outliers.
One can also retain directional information via the signed mean

$$\widehat{\Delta}_{g,d,k}^s \;=\; \mathrm{mean}_{a \in \mathcal{A}_k^s(m)}\, \Delta_{g,d,k,a}^s, \tag{4}$$

indicating whether typical changes to knob k tend to improve or degrade performance on (g, d).

Because $\mathcal{A}_k^{\mathrm{BP}}(m)$ is family-specific, $\Delta_{g,d,k}^{\mathrm{BP}}$ should be interpreted primarily as stability within each family's recommended operating range rather than as a perfectly matched cross-family quantity.

3.5 Preprocessing Sensitivity Index

The *preprocessing sensitivity index* summarizes how sensitive a given model g is to a particular operation type k, across datasets.

Definition 1 (Preprocessing Sensitivity Index)**.** *For a model g, operator space s, and knob $k \in \mathcal{K}$, the (preprocessing) sensitivity index of g to k is*

$$\mathrm{Sens}_k^s(g) \;=\; \mathrm{mean}_{d\in\mathcal{D}}\big(\Delta_{g,d,k}^s\big), \tag{5}$$

where $\Delta_{g,d,k}^s$ is defined in equation 3.

Intuitively, $\mathrm{Sens}_k^s(g)$ answers the question: "On a typical dataset, by how much does the performance of model g change when we alter knob k while keeping the rest of the preprocessing pipeline fixed?". Large values of $\mathrm{Sens}_k^s(g)$ indicate that knob k is highly influential for model g: different reasonable choices for k can produce substantial performance changes. Conversely, small values suggest that model g is comparatively insensitive to that aspect of preprocessing.

A signed variant,

$$\mathrm{DirSens}_k^s(g) \;=\; \mathrm{mean}_{d\in\mathcal{D}}\, \widehat{\Delta}_{g,d,k}^s, \tag{6}$$

captures the typical direction of the effect (helpful vs harmful changes).

3.6 Model Volatility Index

While $\mathrm{Sens}_k(g)$ provides a knob-wise profile of sensitivity for each model, we may also wish to summarize how preprocessing-sensitive a model is overall.

Definition 2 (Model Volatility)**.** *For a model g, its volatility with respect to preprocessing is*

$$\mathrm{Vol}^s(g) \;=\; \mathrm{mean}_{d\in\mathcal{D},\,k\in\mathcal{K}}\big(\Delta_{g,d,k}^s\big). \tag{7}$$

$\mathrm{Vol}^s(g)$ aggregates over both datasets and operation types: it is the mean absolute performance change taken over the joint distribution of all datasets and preprocessing knobs. It represents the expected shock to the system when an arbitrary preprocessing choice is altered; i.e., a high volatility value means that changing one operation induces a performance change for model g. A low volatility value indicates that g is comparatively robust to variations in preprocessing. Because volatility is unsigned, high values may reflect either harmful fragility or exploitable flexibility; the signed metric $\mathrm{DirSens}_k^s(g)$ helps distinguish these cases.

Family-level Aggregation. While the indices above evaluate a specific model g, our primary goal is to characterize the robustness of entire architectural paradigms. We obtain the family-level indices for $m \in \mathcal{M}$ by simply averaging the metrics of its constituent models:

$$\mathrm{Sens}_k^s(m) \;=\; \mathrm{mean}_{g\in\mathcal{G}_m}\, \mathrm{Sens}_k^s(g) \quad \text{and} \quad \mathrm{Vol}^s(m) \;=\; \mathrm{mean}_{g\in\mathcal{G}_m}\, \mathrm{Vol}^s(g). \tag{8}$$

Ultimately, our Volatility Index provides a single quantitative metric to verify claims of 'out-of-the-box' utility and operational robustness across any model family. Sens and Vol are inspired by global sensitivity analysis (Saltelli, 2008): they summarize how changes in preprocessing choices affect an output of interest.

4 Preprocessing Robustness Evaluation Framework

The sensitivity and volatility indices enable a standardized *evaluation framework* for questions such as: *Does a given TFM reduce preprocessing fragility compared to XGBoost or MLP-style models?* We outline the *Preprocessing Robustness Evaluation Framework* (**PREF**), which treats a model family $m \in \mathcal{M}$ as a black box exposing only `fit`/`predict` and outputs knob-wise sensitivity profiles $\{\mathrm{Sens}_k^s(m)\}_{k\in\mathcal{K}}$ and an overall volatility score $\mathrm{Vol}^s(m)$. PREF consists of four elements.

Table 2: Standardized preprocessing knob families used in PREF. Family-specific best-practice subsets defined in Appendix A.

Knob	Representative choices
Numeric scaling / encoding	none, standardization, power scaling, smooth clipping, PLE encoding
Categorical encoding	one-hot, ordinal, target, feature tokenization
Text encoding	TF-IDF, LM embeddings (fastText, E5-style)
Dimensionality reduction	none, truncated SVD, PCA, random projections
Feature selection	none, shuffle-based, Shapley-based
Feature augmentation	first 5 PCA/SVD components, log transforms, squared values

1. Datasets. A curated collection of datasets $\mathcal{D}$ spanning diverse regimes: (i) small vs. large sample size, (ii) low vs. high dimensionality, (iii) different proportions and cardinalities of categorical features, and (iv) with categorical, numerical and textual features.

2. Knobs. A standardized preprocessing knob space $\mathcal{K}$ with admissible choices $\{\mathcal{A}_k\}_{k\in\mathcal{K}}$, shared across model families and instantiated through family-specific best-practice subsets when needed. Table 2 summarizes the knob families considered by PREF and representative alternatives. We leave missing-value imputation and class-imbalance handling to future work, as they introduce model- and objective-dependent choices beyond feature representation (Van Buuren & Groothuis-Oudshoorn, 2011; Josse et al., 2024).

2.a Numerical transformations. For numerical features, classical scaling remains important for optimization, especially for MLP-style models, whereas encodings such as PLE can increase expressivity for deep architectures (Gorishniy et al., 2022). Tree ensembles are often treated as insensitive to monotone transforms, but outliers and depth-limited splits can still make preprocessing consequential in practice (Friedman, 2001). Including numeric scaling and encoding knobs lets PREF test these family-specific assumptions directly.

2.b Category encoders. The choice of input encoding is a first-order source of variation in heterogeneous tables. One-hot and ordinal mappings remain standard categorical baselines (Micci-Barreca, 2001; Ke et al., 2017), while feature tokenization provides a natural interface to transformer-style tabular models (Wang & Sun, 2022). As tables increasingly contain free-text cells, choosing between sparse lexical representations and frozen language-model embeddings becomes equally important (Mráz et al., 2025; Shysheya et al., 2025). These alternatives differ in the balance between lexical matching and semantic abstraction (Agirre et al., 2014), and recent strong text encoders such as E5-style models further enlarge the design space (Wang et al., 2024). Evaluating robustness to categorical and text encodings is therefore central in our setting.

2.c Feature-space contraction and expansion. Feature selection and dimensionality reduction trade off statistical efficiency, computation, and representation size, but they may also discard interactions that some families would otherwise exploit (Kohavi & John, 1997). Conversely, feature augmentation via low-rank components, logarithmic transforms, or squared terms expands the representation and tests whether models can benefit from engineered global structure or nonlinear scale cues. Together, these knobs cover the main representation changes used in practice while being modular enough for single-knob ablations.

3. Pipelines. A reference-pipeline policy $P^{\text{ref}}_{m,d}$ (standardized or expert-crafted, as in the previous section), and a fixed evaluation protocol (metric, splits, number of seeds) for each (m, d).

4. Operator-space. An operator-space policy $s \in \{\text{WC}, \text{BP}\}$ that determines the admissible alternatives $\mathcal{A}^s_k(m)$; PREF defaults to WC (wrong-choice penalties) and reports BP as a complementary view.

The benchmark exposes a simple interface: a user supplies a model $g \in \mathcal{G}_m$ (e.g., an implementation of XGBoost, a deep tabular architecture, or a TFM) and the benchmark returns the metrics in Section 3 (F, Δ, $\widehat{\Delta}$) and ultimately the summary indices $\text{Sens}^s_k(g)$ and $\text{Vol}^s(g)$. Family-level metrics for m are also returned upon completion of the benchmark.

In this work we primarily report results under a lightweight retuning regime, i.e., we re-run a small HPO budget after each knob change for trainable models. We deliberately adopt this regime because it reflects realistic practitioner workflows: in production settings, a preprocessing change is invariably followed by at

least a lightweight re-optimization pass. Measuring sensitivity under frozen hyperparameters would capture an artificial failure mode, the inability to adapt stale configurations, rather than the preprocessing effect itself. Our regime instead isolates the question practitioners face in practice: *given a fixed, modest tuning budget, how much does my model's performance depend on upstream preprocessing choices?*

5 Models, configurations, and evaluation protocol

We detail the model families, their configurations, and the evaluation protocol used for our study.

Model families and concrete instantiations. We evaluate three families of tabular learners: (i) *tree-based gradient-boosted ensembles* (GBDT), (ii) *MLP-style tabular networks*, and (iii) *tabular foundation models* (TFMs) operating in inference-only. We instantiate the families with the following representatives. **Tree-based:** XGBoost, CatBoost, LightGBM; **MLP-style:** RealMLP (Holzmüller et al., 2025), TabM (Gorishniy et al., 2025), ModernNCA (Ye et al., 2025); and **TFM:** TabICL, TabPFN, CTBM (API access).

TFMs' internal preprocessing. TFMs include a non-trivial *internal* preprocessing stack that maps user tables into the representation used during pretraining (details in Appendix C). Thus, dataset-level preprocessing choices in $\mathcal{K}$ compose with this internal transform and can induce distribution shifts relative to the model's pretrained operating point. Under this view, $\mathcal{K}$ probes *compatibility* between practitioner-style upstream pipelines and each TFM's learned input interface. This double-preprocessing reflects realistic enterprise deployments where data often arrives sanitized or encoded by upstream engineering. The fact that TFMs might degrade when fed standard best-practice data is a deployment risk that our framework quantifies.

Training and inference configuration. We separate *model-specific* procedures (architecture details, optimizer, early stopping, prompt/context choices, etc.) from *model-agnostic* preprocessing knobs.

Hyperparameter regime. In this work, we primarily report results under an adaptation capacity (retuning) regime: to measure the true adaptation capacity of each family to representation changes, rather than strict operational robustness without retuning. After each preprocessing change, trainable models re-run a lightweight HPO procedure (e.g., random search with a budget of 10 trials) and are then retrained with the selected hyperparameters. For TFMs, the pretrained backbone remains frozen; only inference-time settings are considered for adaptation when applicable (e.g., number of shots / context size).

Train/test protocol. For each dataset, we evaluate every (model, preprocessing) configuration on 5 independent 80/20 splits generated with seeded *StratifiedShuffleSplit*. For each split, trainable models are initialized and fitted with a fixed model seed. HPO is performed *within the training portion of each split* using validation AUC to select the best trial; the selected hyperparameters are retrained on the full training fold and evaluated on the held-out fold. TFMs follow the same split protocol; prompt/context settings are kept fixed. We report uncertainty as 95% confidence intervals computed using the Student's t-distribution with 4 degrees of freedom across the 5 stratified shuffle folds.

Reference pipelines. Sensitivity values compare single-knob ablations against a chosen reference pipeline. We use three reference conventions. **Classic:** the reference pipeline is a simple default based on `tfidf` for text, one-hot encoding for categorical features, and standardized numerical values. **Model-wise Classic:** the default pipeline depends on the model family: (i) tree-based: `tfidf` + ordinal + standard scaling; (ii) MLP-style: `tfidf` + one-hot + no scaling; (iii) TFMs: `tfidf` + ordinal + no scaling. **Model-wise Best:** the reference pipeline is the best pipeline *for the given model* from experiments.

Datasets. We evaluate pipelines on 7 publicly available datasets, chosen to represent complex and heterogeneous real-world data. They all contain different types of features (*categorical, numerical, text*) and correspond to real classification tasks (e.g., *music genre recognition*):

- **Kickstarter** (Mráz et al., 2025): 19 features, 24k rows; predicts Kickstarter project success.
- **Spotify** (Mráz et al., 2025): 20 features, 24k rows; predicts music genre.
- **Consumer Complaints** (Mráz et al., 2025): 17 features, 24k rows; predicts response to complaints.

- **Job Fraud** (Mráz et al., 2025): 17 features, 24k rows; predicts whether job offers are real or fake.
- **AI vs Human** (Rajak, 2025): 11 features, 500 rows; identifies AI- or human-generated content.
- **CLEAR** (Heintz et al., 2022): 39 features, 4.7k rows; predicts content type (literature or information).
- **Job AI** (Yadav, 2025): 17 features, 3k rows; predicts whether jobs are likely to be automated.

Evaluation metrics and aggregation. We evaluate predictive performance using **accuracy**. As our indices are based on *deltas* rather than absolute scores, accuracy differences faithfully capture the magnitude of preprocessing-induced shifts without the threshold or calibration dependencies introduced by ranking-based (AUC) or probability-based metrics. Extending the analysis to complementary metrics is a natural direction for future work. To aggregate results across datasets, we also report a **dataset-normalized** score rescaled to $[0, 1]$ per dataset (Eq. 1). For signed (*oriented*) sensitivity, the sign indicates whether typical preprocessing changes are beneficial (positive) or harmful (negative) for the considered model family.

6 Results and Analysis

We organize our findings into two parts: first, we analyze the preprocessing characteristics of the different model families (Model Analysis); second, we assess the stability and reliability of the proposed indices themselves (Metric Validation). Unless otherwise stated, results in this section rely on the *Model-wise Classic* reference pipeline to reflect realistic usage scenarios.

6.1 Model Analysis: The Impact of Preprocessing

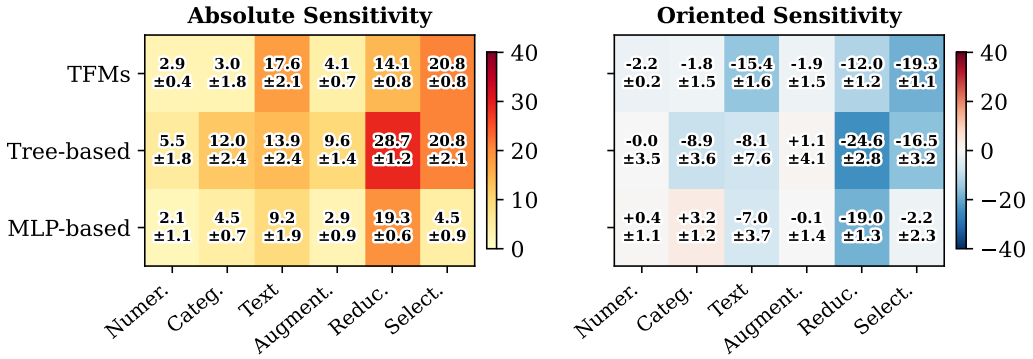

Figure 1: **Family-level Sensitivity.** Heatmap colors highlight family-dependent sensitivity: TFMs are most sensitive to text encoding variations, whereas tree-based and MLP-style models are most sensitive to dimensionality reduction.

Q1: How does preprocessing sensitivity vary across model families? Figure 1 shows that all model families are primarily affected by preprocessing knobs that directly control the number of features—text encoding, feature reduction, and feature selection. These consistently produce the largest sensitivities across families. In contrast, feature augmentation and categorical encoding have a much smaller impact overall, suggesting that most models absorb the added dimensionality through implicit selection mechanisms.

Tree-based models stand out as the most sensitive overall, with strong responses not only to feature count (especially selection and reduction) but also to categorical and text encoding. Numerical preprocessing, however, has almost no effect on them, confirming their invariance to scaling.
MLP-style models are globally less sensitive, but their sensitivity is more concentrated: they react primarily to feature reduction and, to a lesser extent, selection and text encoding, while being relatively stable to categorical encoding and augmentation.
TFMs show a different pattern: their sensitivity is more evenly spread, with clear impact from text encoding and feature selection, but still limited response to augmentation and numerical preprocessing.

These trends are statistically robust: the confidence intervals in Figure 2 are narrow compared to inter-family differences. Overall, the figure shows that preprocessing sensitivity is structured along two axes: (i) whether a knob alters the effective feature space, and (ii) how each model family handles representation and selection.

Contrary to the promise of full, "out-of-the-box" automation, TFMs do not abstract away the need for preprocessing, nor do they exhibit uniformly low volatility. Instead, they display a hybrid behavior where sensitivity reflects both architectural biases and the compatibility between upstream preprocessing and the data distribution encountered during pretraining. TFMs show fragility when upstream transforms, such as specific text encoders, interact adversely with their internal distribution assumptions, shifting the data out of the pretrained operating point. Ultimately, preprocessing remains a critical design choice across all families.

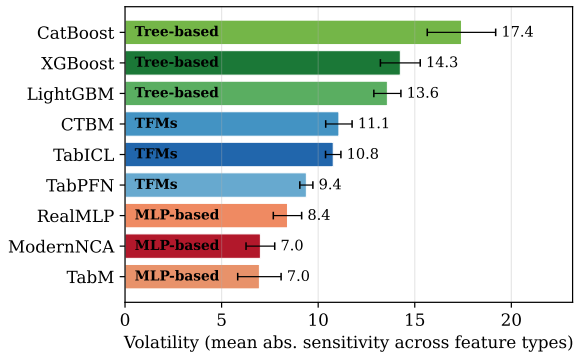

Figure 2: **Model Volatility Ranking**. Mean absolute performance across knobs and datasets under $\mathcal{A}^{\text{WC}}$ (space of common, possibly mismatched preprocessing choices). Lower is more robust; higher means stronger preprocessing dependence.

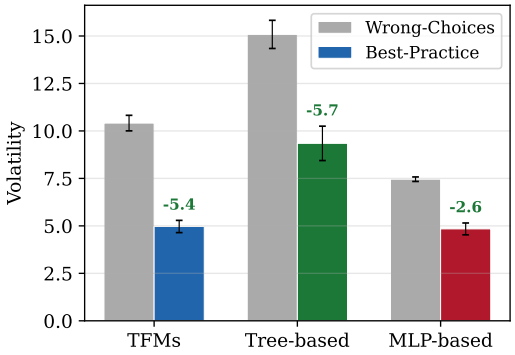

Figure 3: **Wrong- vs Best-Practice Volatility**. A large drop from $\mathcal{A}^{\text{WC}}$ (common, possibly mismatched preprocessing choices) to $\mathcal{A}^{\text{BP}}$ (only family-compatible best practices) means sensitivity is mostly avoidable user error; sensitivity persists even under recommended preprocessing.

Q2: How much volatility is driven by "user error" versus within-family robustness? Figure 3 quantifies the volatility reduction when restricting preprocessing choices from *Wrong-Choice* ($\mathcal{A}^{\text{WC}}$) to *Best-Practice* ($\mathcal{A}^{\text{BP}}$). Tree-based models exhibit a large reduction, indicating that a substantial portion of their volatility arises from avoidable out-of-range choices (e.g., ill-posed encodings under high cardinality). In contrast, TFMs show a smaller reduction under the same restriction, suggesting that a non-trivial component of their sensitivity persists even within curated best-practice alternatives.

Q3: Is high volatility harmful? Not necessarily: high sensitivity can signal tuning *opportunity* rather than fragility. Figure 4 compares each model's volatility against its tuning gain (relative improvement from optimizing preprocessing over the default pipeline). Tree-based models combine moderate volatility with substantial tuning gains ($\sim$4% for XGBoost), indicating that their sensitivity is largely exploitable through feature engineering. MLP-style models cluster in the low-volatility/low-gain region. TFMs occupy an intermediate position: their rigid internal preprocessing restricts upstream optimization gains while maintaining moderate volatility. The oriented sensitivity (DirSens) distinguishes these cases per knob: a knob with high absolute sensitivity and positive direction is an opportunity, while one with negative direction is a risk.

Q4: Are these patterns consistent across datasets? While family-level rankings remain stable, preprocessing sensitivity is dataset-dependent (Figure 5). However, Figure 10 (Appendix B) shows that family-level ranks stabilize after 4–5 datasets, suggesting our 7-dataset suite is sufficient for robust conclusions.

6.2 Metric Validation: Stability of the Indices

Q5: Is the metric sensitive to the choice of the reference pipeline? A robust index should not depend heavily on the arbitrary choice of a baseline. Figure 6 demonstrates that while absolute sensitivity

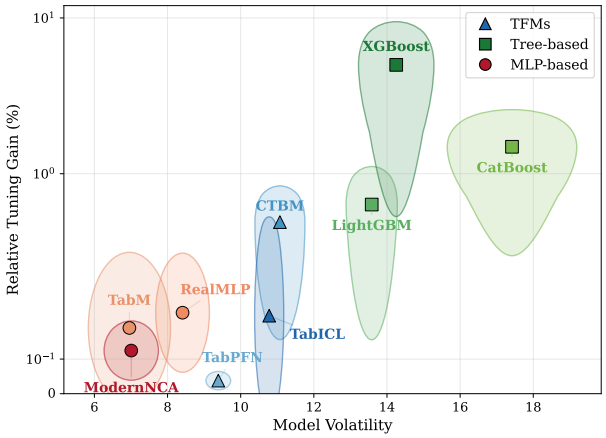

Figure 4: **Model sensitivity vs Improvement**. Model-wise improvement capacity (average difference between default and best performance across datasets and knobs) compared to its sensitivity.

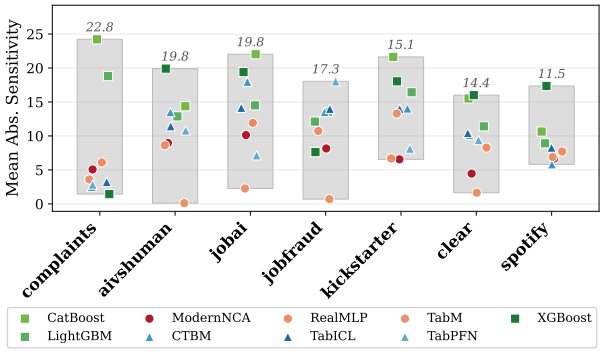

Figure 5: **Dataset-level sensitivity spread**. Models' mean absolute sensitivity per dataset; grey bands span the min–max range across models. Datasets sorted by decreasing cross-model range.

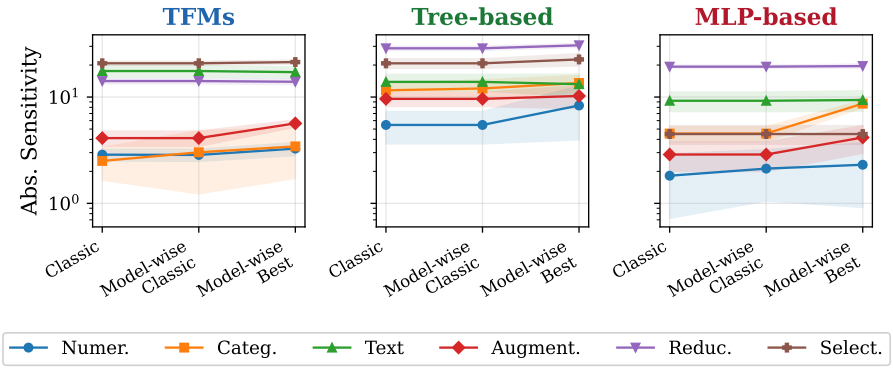

Figure 6: **Sensitivity across reference pipelines**. Absolute preprocessing sensitivity for each knob under three regimes. Panel shows a model family, colors are preprocessing types, and shaded regions indicate variability. The near-parallel trends across regimes indicate that sensitivity rankings are stable, suggesting that preprocessing effects are intrinsic to model families rather than artifacts of the chosen reference pipeline.

values shift slightly between reference policies (e.g., Global Classic vs. Model-wise Best), the relative rankings and key sensitivity patterns (the "fingerprint" of the model family) remain comparatively stable.

Q6: Does the metric distinguish between operator spaces effectively? Our results confirm that the indices successfully capture the contraction in variance when moving from the full operator space to the best-practice subset (Figure 3). This validates the design choice of reporting Δ^{WC} and Δ^{BP} separately to distinguish operational robustness and error tolerance. As detailed in our analysis (Appendix B, Figure 11), the *Preprocessing Sensitivity* dominates the cross-validation standard deviation across folds. This validates that our metric captures true responsiveness to representation changes rather than random evaluation variance.

7 Practical Guidelines and Robustness Cards

The Preprocessing Robustness Evaluation Framework (PREF) moves beyond simple accuracy leaderboards. By treating the model as a black box and systematically ablating data transformations, the framework

generates a *"Robustness Card"* for each model family. This card functions as a safety manual, delineating the safe operating range of data distributions and preprocessing choices under which the model remains reliable. We provide representative examples of these cards in Table 3 and detail their components below.

Table 3: **Example Robustness Cards** for representative models. **Vol**: volatility under user-error/default choices. **Gap**: reduction in volatility when restricting to best practices. **Critical**: knobs causing largest drops. **Expected Payoff**: knobs yielding highest gains.

Family	Model	Vol	Gap	Critical Knobs (Risk)	Expected Payoff (Tune)
Tree-based	XGBoost	High	Large	Categorical (High-Card.)	Feature Augmentation
MLP-style	TabM	Low	Small	Numerical Scaling	Text Embeddings
TFM	TabICL	Med	Small	Scaling, Text Enc.	Context/Serializer

The card provides actionable guidelines that support deployment decisions through:

- **Volatility Index:** A global stability score ($\mathrm{Vol}^{\mathrm{WC}}$). Models with High volatility (e.g., XGBoost) see large performance swings across pipelines, signaling high tuning opportunity. Models with Low volatility (e.g., TabM) are comparatively stable on average. TFMs display Medium volatility (e.g., TabICL), which requires adherence to specific input formats and contradicts the "AutoML" promise.
- **The Sensitivity Gap:** This metric quantifies tolerance to wrong choices ($\mathrm{Vol}^{\mathrm{WC}} - \mathrm{Vol}^{\mathrm{BP}}$).
  - Large Gap (e.g., Tree-based): The model performs well under best practices but fails catastrophically with naive choices (e.g., one-hot encoding a high-cardinality column). This indicates the model is robust *conditional* on basic engineering competence.
  - Small Gap (e.g., TFMs): The model's instability is inherent to its architecture and pretraining; constraining choices to best practices has small returns in stability.
- **Critical Knobs (Risk):** Operations with large impact. For MLP-style models, omitting Numerical Scaling is a critical failure mode. For TFMs, mismatches in Text Encoding or Scaling often conflict with internal distribution shifts, degrading performance below simple baselines.
- **Expected Payoff (Tune):** Knobs where sensitivity is high and *oriented* effect is positive. These are the axes where domain expertise (feature engineering) yields the highest return on investment.

We will release the PREF toolkit as an open-source library. Researchers proposing new tabular architectures can use it to generate these cards automatically, ensuring that future claims of robustness and "ease of use" are substantiated by empirical volatility metrics rather than anecdotal evidence.

8 Conclusion

We studied preprocessing as a first-class source of variation in heterogeneous tabular learning by organizing common transformations into a knob space and running controlled single-knob ablations across three model families. We introduced measures to summarize knob-wise and aggregate sensitivity to preprocessing choices, and packaged the resulting measurements into Robustness Cards for practitioners. Across text-rich datasets, we find that preprocessing sensitivity is strongly model-family-dependent: trees exhibit the highest aggregate volatility, MLPs show lower overall volatility but possess critical vulnerabilities to specific representation shifts, and TFMs are not uniformly robust, with pronounced sensitivity to text and categorical handling.

Limitations. Our study is limited to seven public classification datasets and a fixed set of preprocessing knobs; we do not evaluate missing-value imputation or class-imbalance handling, and we focus on accuracy as the performance metric. Sensitivities are reported under a lightweight retuning regime with a fixed HPO budget, which reflects practical adaptation capacity but may differ from strict fixed-hyperparameter operational robustness. Extending the dataset suite, task types, and metric coverage is left to future work.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pp. 81–91, 2014.

Furkat Bolikulov, Rashid Nasimov, Akbar Rashidov, Farkhod Akhmedov, and Young-Im Cho. Effective methods of categorical data encoding for artificial intelligence algorithms. *Mathematics*, 12(16), 2024. ISSN 2227-7390. doi: 10.3390/math12162553. URL https://www.mdpi.com/2227-7390/12/16/2553.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(6):7499–7519, 2024. doi: 10.1109/TNNLS.2022.3229161.

Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002. URL https://jmlr.org/papers/v2/bousquet02a.html.

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

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Mutalik Desai, David Salinas, and Frank Hutter. Tabarena: A living benchmark for machine learning on tabular data, 2025. URL https://arxiv.org/abs/2506.16791.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29 (5):1189–1232, 2001.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *arXiv preprint arXiv:2106.11959*, 2021.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. In *NeurIPS*, 2022.

Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. Tabm: Advancing tabular deep learning with parameter-efficient ensembling, 2025. URL https://arxiv.org/abs/2410.24210.

Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. 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*, 2022. URL https://openreview.net/forum?id=Fp7__phQszn.

Aron Heintz, Joon Suh Choi, Jordan Batchelor, Mehrnoush Karimi, and Agnes Malatinszky. A large-scaled corpus for assessing text readability. *Behavior Research Methods*, 55, 03 2022. doi: 10.3758/s13428-022-01802-x.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned mlps and boosted trees on tabular data, 2025. URL https://arxiv.org/abs/2407.04491.

Julie Josse, Jacob M. Chen, Nicolas Prost, Erwan Scornet, and Gaël Varoquaux. On the consistency of supervised learning with missing values, 2024. URL https://arxiv.org/abs/1902.06931.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pp. 3146–3154, 2017.

Myung Jun Kim, Léo Grinsztajn, and Gaël Varoquaux. Carte: Pretraining and transfer for tabular learning, 2024. URL https://arxiv.org/abs/2402.16785.

Aaron Klein and Frank Hutter. Tabular benchmarks for joint architecture and hyperparameter optimization. *CoRR*, abs/1905.04970, 2019. URL http://arxiv.org/abs/1905.04970.

Ron Kohavi and George H John. Wrappers for feature subset selection. In *Artificial intelligence*, volume 97, pp. 273–324. Elsevier, 1997.

Kyungeun Lee, Moonjung Eo, Hye-Seung Cho, Dongmin Kim, Ye Seul Sim, Seoyoon Kim, Min-Kook Suh, and Woohyung Lim. Multitab: A comprehensive benchmark suite for multi-dimensional evaluation in tabular domains, 2025. URL https://arxiv.org/abs/2505.14312.

Si-Yang Liu, Hao-Run Cai, Qi-Le Zhou, and Han-Jia Ye. Talent: A tabular analytics and learning toolbox, 2024. URL https://arxiv.org/abs/2407.04057.

Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak Prasad C., Ganesh Ramakrishnan, Micah Goldblum, and Colin White. When do neural nets outperform boosted trees on tabular data? In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Daniele Micci-Barreca. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter*, 3(1):27–32, 2001.

Martin Mráz, Breenda Das, Anshul Gupta, Lennart Purucker, and Frank Hutter. Towards benchmarking foundation models for tabular data with text, 2025. URL https://arxiv.org/abs/2507.07829.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: unbiased boosting with categorical features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 6639–6649, 2018.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data, 2025. URL https://arxiv.org/abs/2502.05564.

Stephan Rabanser, Tim Januschowski, Valentin Flunkert, David Salinas, and Jan Gasthaus. The effectiveness of discretization in forecasting: An empirical study on neural time series models. *arXiv preprint arXiv:2005.10111*, 2020.

Prince Rajak. Ai vs human comparison dataset. https://www.kaggle.com/datasets/prince7489/ai-vs-human-comparison-dataset, 2025. Kaggle dataset. Side-by-side comparison of AI-generated and human-written text across multiple topics.

Ivan Rubachev, Nikolay Kartashev, Yury Gorishniy, and Artem Babenko. Tabred: Analyzing pitfalls and filling the gaps in tabular deep learning benchmarks, 2024. URL https://arxiv.org/abs/2406.19380.

A. Saltelli. *Global sensitivity analysis: the primer*. John Wiley, 2008. ISBN 9780470059975. URL http://books.google.at/books?id=wAssmt2vumgC.

Aliaksandra Shysheya, John Bronskill, James Requeima, Shoaib Ahmed Siddiqui, Javier Gonzalez, David Duvenaud, and Richard E. Turner. Jolt: Joint probabilistic predictions on tabular data using llms, 2025. URL https://arxiv.org/abs/2502.11877.

Marco Spinaci, Marek Polewczyk, Maximilian Schambach, and Sam Thelin. Contexttab: A semantics-aware tabular in-context learner. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=kGMRb4jbTP.

Andrej Tschalzev, Sascha Marton, Stefan Lüdtke, Christian Bartelt, and Heiner Stuckenschmidt. A data-centric perspective on evaluating machine learning models for tabular data. *Advances in Neural Information Processing Systems*, 37:95896–95930, 2024.

Stef Van Buuren and Karin Groothuis-Oudshoorn. Mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training, 2024. URL `https://arxiv.org/abs/2212.03533`.

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

Khushi Yadav. Ai impact on jobs 2030. `https://www.kaggle.com/datasets/khushikyad001/ai-impact-on-jobs-2030/data`, 2025. Kaggle dataset. Simulates the future of work in the age of artificial intelligence. It models how various professions, skills, and education levels might be impacted by AI-driven automation by the year 2030.

Han-Jia Ye, Huai-Hong Yin, De-Chuan Zhan, and Wei-Lun Chao. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=JytL2MrlLT`.

A Operator Spaces

Choices for Best-Practice alternatives ($\mathcal{A}_k^{\mathrm{BP}}$) were made according to commonly used and documented practices and our own knowledge on data preprocessing. We report below the selected methods per knob and model-family.

- **Tree-based models** *numerical*: [none, standard, power], *categorical*: [ordinal, target], *text*: [tfidf, E5 small v2], *augmentation*: [none, pca, svd], *reduction*: [none, pca (explained variance), svd], *selection*: [none, shuffle]
- **MLP-style models** *numerical*: [standard, power, PLE], *categorical*: [ordinal, one-hot, target], *text*: [tfidf, E5 small v2], *augmentation*: [none, log, squared], *reduction*: [none, pca (explained variance), svd], *selection*: [none, shuffle]
- **TFMs** *numerical*: [none, standard, power], *categorical*: [ordinal, target], *text*: [tfidf, E5 small v2], *augmentation*: [none, pca, svd], *reduction*: [none, pca (explained variance), svd], *selection*: [none, shuffle]

Choices for augmentation knob on tree-based models are the same as for the TFMs. Indeed, since few augmentation methods are commonly used for this model family, we chose to follow the same settings.

B Additional Analyses and Validation

B.1 Downward risk vs uplift potential

For each model, we collect all signed normalized deltas (relative to the model-wise classic reference) across every dataset and feature type. Positive deltas — cases where an alternative preprocessing method outperforms the reference — are separated from negative ones. We summarize each side by its median: the *upside potential* is the median positive delta, while the *downside risk* is the median absolute negative delta. Family-level values are obtained by averaging member-model medians. Points above the diagonal indicate models for which the typical gain from exploring alternative preprocessing outweighs the typical loss, making experimentation worthwhile.

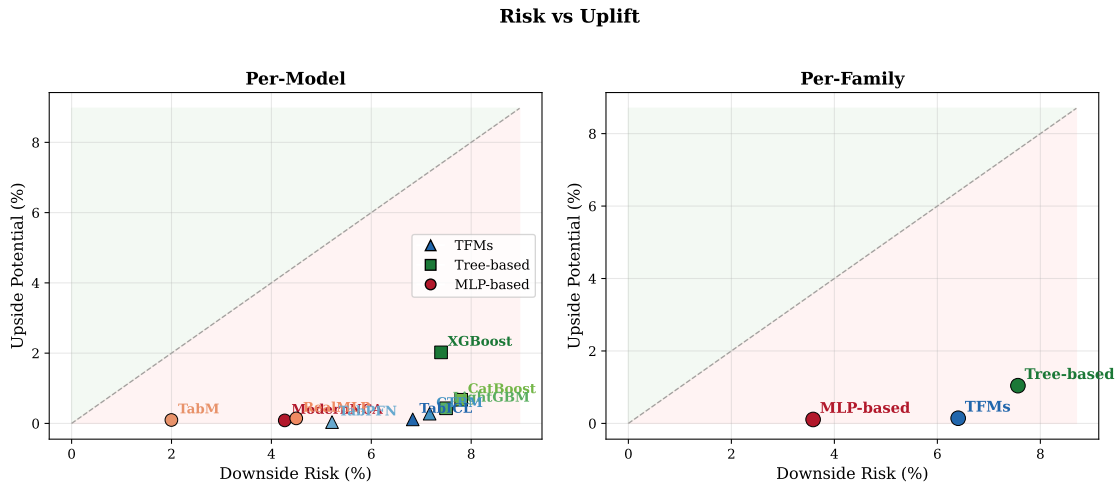

Figure 7: **Upside potential vs. downside risk per model (left) and per family (right).** Each axis shows the median normalized delta (%) for methods that improve (y-axis) or degrade (x-axis) accuracy relative to the model-wise classic reference. Points above the diagonal correspond to models for which trying alternative preprocessing is, on balance, beneficial.

B.2 Computing sensitivity per dataset

Whereas the global sensitivity analysis averages deltas across all datasets, here we break it down: for each combination of dataset, model, and feature type, we compute the mean absolute (or oriented) normalized delta with respect to the model-wise classic reference, then aggregate to the family level by averaging over member models. This reveals dataset-specific patterns that the global view may mask — for instance, a family may appear robust overall yet exhibit high sensitivity on a particular dataset.

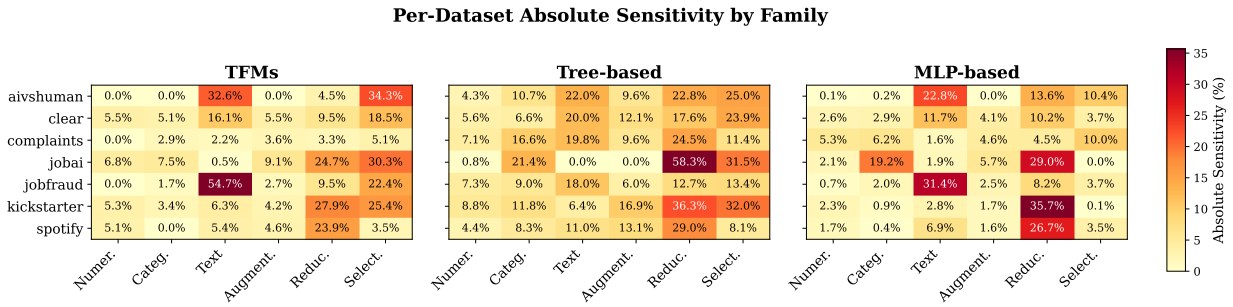

Figure 8: **Per-dataset absolute sensitivity by family.** Each heatmap shows one model family; rows are datasets, columns are feature types, and cell values are the mean absolute normalized delta (%) relative to the model-wise classic reference. Higher values (darker cells) indicate greater sensitivity to preprocessing choices for that dataset–feature combination.

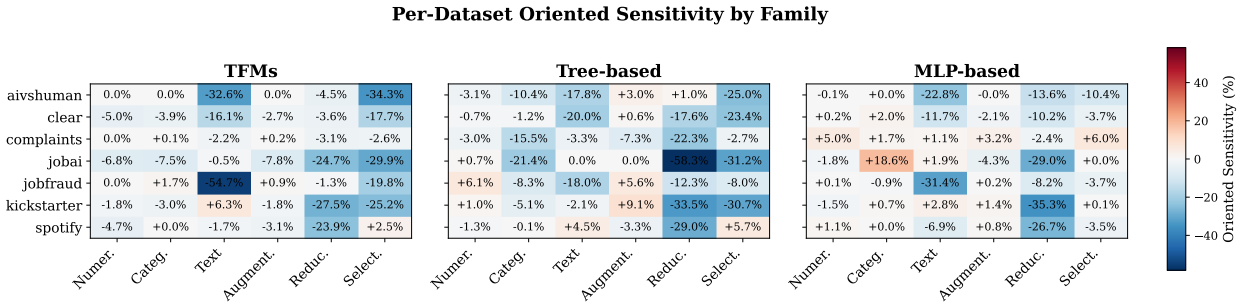

Figure 9: **Per-dataset oriented sensitivity by family.** Same layout as Figure 8, but cells show the mean signed normalized delta (%). Positive values (blue) indicate that alternative methods tend to improve over the reference, while negative values (red) indicate they tend to degrade performance.

B.3 Sensitivity ranking analysis

To assess whether the robustness ordering of models is consistent across datasets, we rank models and families by their per-dataset volatility (mean absolute sensitivity across feature types), where rank 1 denotes the least volatile (most robust). These per-dataset ranks are displayed alongside the global rank derived from the full cross-dataset analysis. Stable lines in the bump chart and uniform rows in the heatmap indicate that a model's robustness is an intrinsic property rather than a dataset-dependent artefact.

B.4 Signal-to-noise analysis

A natural question is whether the observed preprocessing sensitivity reflects a genuine effect or is merely an artifact of cross-validation variance. To answer this, we compare two quantities computed on the same scale (raw accuracy points): the *signal*, defined as the standard deviation of accuracy across preprocessing methods for a given dataset–feature pair, and the *noise*, defined as the mean standard deviation of accuracy

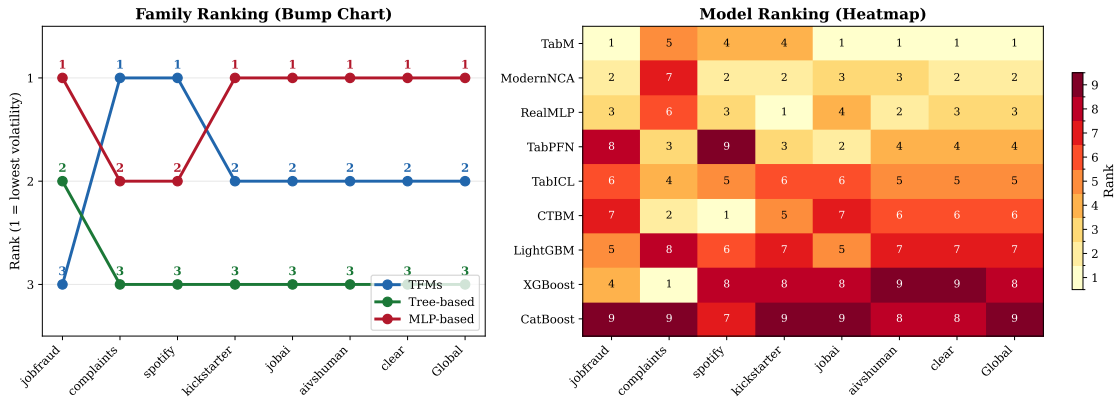

Figure 10: **Ranking stability across datasets.** Left: bump chart of family ranks (1 = most robust); right: heatmap of model ranks across datasets and globally. A model whose rank varies little across columns exhibits stable robustness regardless of the dataset.

across the five cross-validation folds (averaged over methods). Both are then averaged per model. Models below the diagonal exhibit a signal-to-noise ratio greater than one, confirming that the preprocessing effect exceeds the stochastic variability introduced by the train/test split.

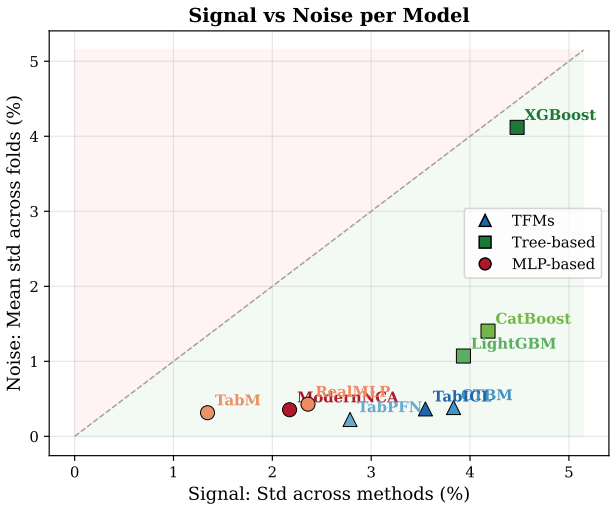

Figure 11: **Signal-to-noise ratio of preprocessing effects.** The x-axis measures the mean standard deviation of accuracy across preprocessing methods (signal), while the y-axis measures the mean standard deviation across cross-validation folds (noise), both in raw accuracy percentage points. Models below the diagonal have signal > noise, indicating that preprocessing differences are larger than the variability due to cross-validation.

C TFMs Preprocessing

Modern **TabPFN** releases include an explicit preprocessing pipeline that converts heterogeneous tables into a normalized representation expected by the transformer. Internally, TabPFN performs (i) feature-type handling (numerical vs. categorical), (ii) distribution shaping / scaling of numerical features, (iii) categorical encoding choices, and (iv) feature-level randomization such as column shuffling as part of the ensemble inference procedure. It also includes optional global transforms that can act as dimensionality control (e.g., truncated-SVD-like projections) and utilities such as constant-feature removal. Consequently, upstream

"numeric feature scaling" or "categorical encoding" knobs in $\mathcal{K}$ should be interpreted as *input distribution interventions* that may either align with, or conflict with, TabPFN's own learned regime.

TabICL and **CTBM** detect categorical columns and apply ordinal encoding, assign a dedicated category to missing values in categorical features, and impute missing numerical values with the mean. Beyond this input sanitation step, TabICL applies additional preprocessing as part of inference-time ensembling, including outlier handling (clipping), feature scaling/normalization, and feature shuffling for ensemble diversity. CTBM performs similar operations but the details are part of its proprietary algorithm and are not visible with API access. Therefore, our knobs for categorical encoding, missing-value imputation, and numeric scaling partially overlap with operations performed internally.