International Journal of Engineering, Science and Information Technology
Volume 5, No. 3 (2025) pp. 448-455
eISSN: 2775-2674
Website: http://ijesty.org/index.php/ijesty
DOI: https://doi.org/10.52088/ijesty.v5i3.1146
Research Paper, Short Communication, Review, Technical Paper

Transformer-Based Tabular Foundation Models: Outperforming Traditional Methods with TabPFN

R Anand Babu[1*], Vishwa Priya V[2], Manoj Kumar Mishra[3], Inakoti Ramesh Raja[4], Surya Kiran Chebrolu[5], B Swarna[6]

[1]Department of AI&ML, Panimalar Engineering College Autonomous, Chennai, India.
[2]Department of Computer Science, Vels Institute of Science, Technology and Advanced Studies, Chennai, India.
[3] Department of Artificial Intelligence & Data Science, Parul Institute of Engineering & Technology, Vadodara, India
[4]Department of Electronics and Communication Engineering, Aditya University, Surampalem, India.
[5]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India.
. [6]University Centre for Research & Development, Chandigarh University, Mohali, India.

**Corresponding author Email: ranandbabu215@gmail.com*

The manuscript was received on 17 January 2025, revised on 15 June 2025, and accepted on 27 June 2025, date of publication 7 July 2025

Abstract

Scientific research and commercial applications rely heavily on tabular data, yet efficiently modelling this data has constantly been a problem. For over twenty years, the standard method for machine learning has been based on traditional models, with gradient-boosted decision trees (GBDTs). Despite recent advancements in deep learning, neural networks often fail to provide satisfactory results on compact tabular datasets due to factors such as overfitting, insufficient data & intricate feature relationships. The study offers a Tabular Prior data Fitted Network, a foundation model developed by meta-learning on more than one million synthetic datasets generated sequentially, which is constructed on transformers to tackle these limitations. Without retraining or hyperparameter optimization, TabPFN learns to anticipate the best solutions for tabular problems, gaining inspiration from the achievements of GPT-like models in natural language processing. When applied to small to medium-sized datasets, its cutting-edge performance in inference speed & accuracy outperforms that of traditional methods. TabPFN redefines efficient and scalable tabular data modelling, including generative capabilities, few-shot learning, & rapid adaptation.

Keywords: *Tabular Data, Deep Learning, Gradient-Boosted Decision Trees, Generative Capability, Few-Shot Learning, Transformer.*

1. Introduction

Various disciplines rely on tabular data, which consists of structured information organized in rows and columns. These subjects include healthcare, finance, physics & materials science among other applications [1]. The ability to make predictions, evaluate risks & uncover new information relies heavily on predictive tasks like classification and regression in these contexts. The benchmark for these tasks has long been tree-based ensemble algorithms like XGBoost, LightGBM & CatBoost because of their outstanding performance on heterogeneous features & small sample sizes [2-5These methods call for significant hyperparameter changing and domain-specific feature engineering, which could be computationally and time-consuming [2]. At the same time, domains like computer vision and natural language processing have been revolutionized by the advent of foundation models, which are big pretrained neural networks like ViT, GPT & BERT. These models often achieve contemporary performance in low-data regimes by learning general-purpose interpretations that transfer across problems with minimum extra training [3].
This paradigm is the basis for our proposed foundation model for tabular data, the Tabular Prior data Fitted Network (TabPFN). Using a combination of Bayesian networks, Gaussian processes & random forests, TabPFN is trained through meta-learning on millions of tiny synthetic datasets [4]. The model can make correct predictions on fresh data in a few-shot or zero-shot situation without needing gradient updates, since it has internalized generic learning techniques via this process [5]. Instead of working with tokens or picture patches, TabPFN uses a transformer architecture modified to process feature-label pair sequences. The model can handle generative tasks like density estimation and data synthesis because of its structure, which also captures complicated inter-feature connections. Especially in situations with little data and variability, TabPFN routinely outperforms deep learning models and conventional machine learning

baselines in tests using typical tabular benchmarks. The study lays out the steps to create TabPFN, dissects its novel architectural features, and compares it to other state-of-the-art methods. By moving away from task-specific learners and toward general-purpose foundation models, TabPFN competes with GBDTs and changes our perspective on tabular modelling [6].

Traditional tabular learning algorithms like XGBoost, LightGBM, and CatBoost have long-standing dominance in structured data challenges, particularly when working with tiny datasets. The key to their success is mastering engineering features and regularization procedures for tabular data [7]. On the other hand, they rely primarily on domain-specific and human hyperparameter changes, which may be computationally costly & time-consuming. On the other hand, models based on neural networks provide the prospect of end-to-end learning via automated feature extraction and pattern modelling. However, they perform poorly in regimes with minimal data since effective generalization requires massive data [2]. By presenting a foundation model technique for handling tabular data, TabPFN meets this requirement. For generalization to new datasets without fine-tuning, TabPFN learns to represent the joint distribution of features & labels by using the capability of pretrained transformers trained via meta-learning on millions of synthetic jobs Performance on small to medium datasets may now be redefined as being viable without compromising on speed or scalability, to this paradigm change that transfers the benefits of foundation models to the tabular domain, including efficiency, robustness & transferability as shown in table 1 [8].

Table 1. Literature Review: Tabular Data Modelling

Study / Model	Year	Approach	Key Features	Strengths	Limitations
XGBoost (Chen & Guestrin)	2016	Gradient Boosted Trees	Boosting over decision trees with regularization	High performance on structured data; handles missing data well	Requires hyperparameter tuning; less efficient for small datasets
LightGBM (Ke et al)	2017	Gradient Boosted Trees	Leaf-wise growth; histogram-based training	Fast training, low memory usage	Can overfit on small data
CatBoost (Prokhorenkova et al.)	2018	Gradient Boosted Trees	Categorical feature encoding; ordered boosting	Handles categorical data natively	Limited generalization beyond specific tabular tasks
TabNet (Arik & Pfister)	2021	Deep Learning	Sequential attention-based feature selection	Interpretability via attention; end-to-end learning	Slower training requires more data
NODE (Popov et al)	2019	Neural Oblivious Decision Ensembles	Ensemble of differentiable decision trees	A hybrid of tree and neural models	Not robust to all tabular distributions
SAINT (Somepalli et al)	2021	Self-Attention for Tables	Transformer architecture with column masking	Better performance than MLPs on classification tasks	Needs large datasets; underperforms on small-data regimes
FT-Transformer (Gorishniy et al)	2021	Transformer for Tabular Data	Feature tokenization and row-wise transformers	Leverages transformers; competitive results	Requires careful tuning
TabPFN (Hollenstein et al.)	2023	Meta-learned Transformer	Pretrained on millions of synthetic tasks; zero-shot prediction	SOTA performance on small data, no tuning, fast inference	

2. Methods

This has become very rampant in the recent past, hence making flooding one of the areas of interest. In the past, the forecast was based on the hydrological models and a statistical approach that was slow and less accurate. Some of the rising trends witnessed in Artificial Intelligence and Machine learning regarding flood prediction are as follows. In [2], it is also stated that one can use AI to analyze the information gained from satellites and climate data in combination with historical records of floods [9].

2.1. Foundation Model for Tabular Data

Several disciplines rely on tabular data, consisting of rows representing samples and columns representing attributes, including physics, healthcare & finance [10]. Although deep learning models such as transformers have shown exceptional ability in processing unstructured data like text & pictures, their application directly to structured, tabular data has always lagged. The Tabular Prior data Fitted Network (TabPFN) closes this gap by applying the foundation model paradigm, a model previously trained on extensive, varied data to be flexible to many tasks, to the field of tabular learning [11].

The main idea is to view the prediction task as a meta-learning challenge in which the model is trained not only to fit data but also to learn from any new tabular dataset. As shown in the Figure, TabPFN is pretrained on a wide range of synthetic tabular datasets, as GPT is pretrained on a diverse corpus of text and can generalize to many language issues. Ranging in number of features, data kinds, distributions & underlying connections, these datasets let TabPFN adopt the inductive biases required for tabular tasks. Once pretrained, TabPFN is a genuine tabular foundation model as it can generalize to unobserved datasets with little or no further training.

Let a tabular dataset be defined as

$$D = \{(x_i, y_i)\}_{i=1}^{n} \quad \text{.....(1)}$$

$x_i \in R^d$ is a feature vector, and y_i is the corresponding label. A standard predictive model learns a function

$$f_0 : R^d \rightarrow R \quad \text{.....(2)}$$

In contrast, TabPFN learns a meta-function:

$$F_0 : D_{strain} \rightarrow f_0 \quad \text{.....(3)}$$

Where F_Φ is trained over many synthetic datasets, learning how to generate a task-specific model f_θ for a new dataset D_{train}, here, Φ represents the parameters of the foundation model (e.g., transformer weights).

To learn a function $f_\theta : R_d \rightarrow Y$ by minimizing a loss over training data:

$$\theta = \arg min \sum_{i=1}^{N} L(f_\theta(x_i), y_i) \quad \text{.....(4)}$$

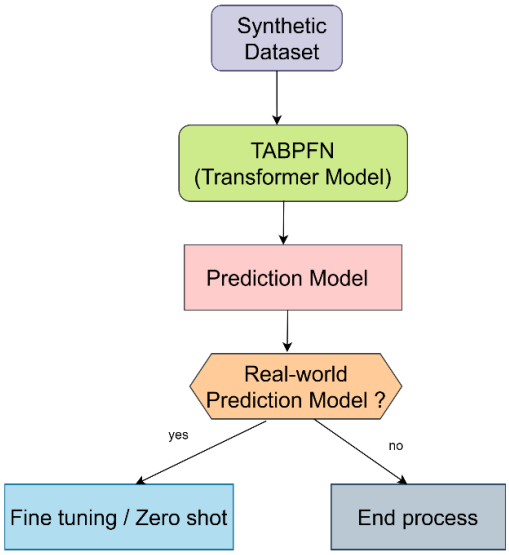

Fig 1. TabPFN workflow model

2.2. Meta Learning Approach

TabPFN's fundamental approach starts with creating millions of synthetic datasets to replicate the complexity and variety of actual tabular issues, as shown in Figure 2 [12]. Ensuring broad coverage of the problems a model may encounter in actual situations, these datasets differ in feature distributions, class counts, correlations & noise levels. Transformer-based foundation models are a generalized learning process, with any new tabular dataset using these fake tasks as input [13]. The model is taught to grasp the fundamental learning structure independently, rather than tackling one task. When the meta-learning phase is complete, the model may quickly generalize to new, unexplored tabular datasets, yielding extremely accurate predictions without conventional restructuring, fine-tuning & hyperparameter optimization techniques. This creates a strong zero-shot learner that significantly lowers the time & computer resources required for tabular data modelling purposes [14].

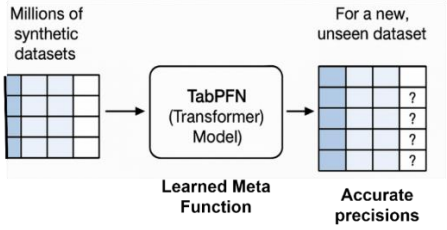

Fig 2. Transformer-Based Foundation Model

2.3. Transformer-based Architecture

In a tabular dataset, rows represent data instances (examples), and columns are features x1,x2,..., xd, and a label y. The data is organized as pairs (x, y) (feature vector & matching label) as shown in Table 3. Millions of synthetic datasets like these are generated during pre-training to mimic real-world categorization activities. In the transformer encoder, an embedding is created from every row of (xi,yi). These embeddings are fed into a Transformer architecture [15]. Models' dependencies between many feature-label pairings using multi-head self-attention. It captures correlations among characteristics and samples, both intra-row & inter-row. The Transformer is designed to turn this list of tabular tokens into predictions. The Transformer gets new data (e.g., feature vectors in masked labels for inference prediction). The model then directly predicts (e.g, class probabilities) for such cases. Rather than training a new model for every dataset, the Transformer generates precise predictions in a single forward pass, avoiding conventional retraining [16].

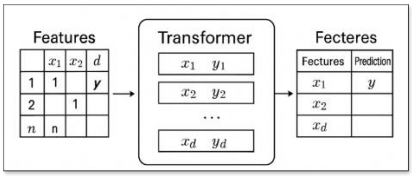

Fig 3. High-level overview of TabPFN pretraining and usage

The architectural diagram shown in Figure 4 describes the TabPFN (Tabular Prior data Fitted Network) pipeline's pretraining and inference workflows. The pipeline generates synthetic tabular data using Bayesian Networks, Gaussian Processes & Random Forests. These models are used to imitate the real-world dataset's statistical diversity. Over a million synthetic datasets with distinct learning tasks are created from them. These datasets underpin meta-learning TabPFN training [17].

The TabPFN Transformer is then trained on simulated datasets. TabPFN implements this architecture for tabular data, unlike text transformers. The processor handles feature-label pairs, not word tokens [18]. The model uses an input encoding layer to convert tabular data rows into vector embeddings [19]. A feature tokenization module encodes each feature and its value as transformer-friendly tokens from these embeddings.

The redesigned self-attention mechanism is a crucial innovation that handles non-sequential and unordered tabular data. This allows the model to capture correlations between characteristics inside and across samples using language model attention patterns tailored for structured inputs. The system enters few-shot inference after the output head translates the final concealed data into predictions. TabPFN may use its pretrained information to quickly guess the unknown label from an entirely new input sequence (e.g., feature-label pairs with one label missing) without retraining or hyperparameter adjustment [11]. This design lets TabPFN learn from many synthetic jobs & adapt to new datasets with little data & compute, making it a strong tabular data foundation model.

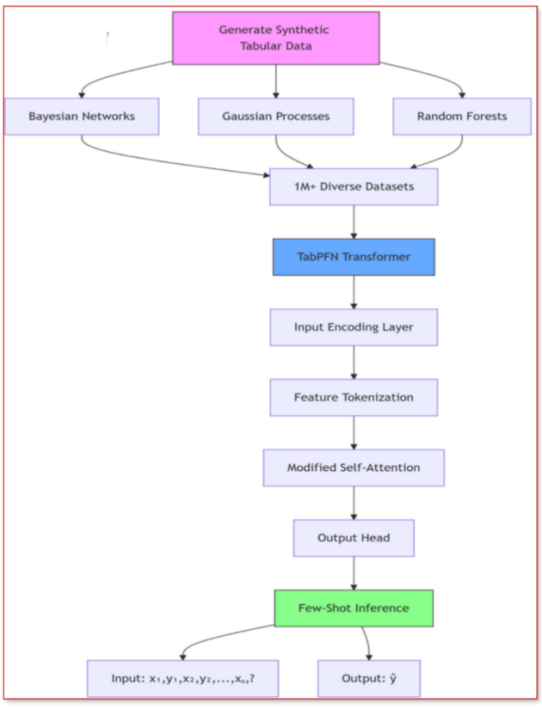

Fig 4. Architectural diagram of TabPFN

2.4. Generative Modelling and Pretraining

Using synthetic data creation, density evaluation & learning reusable embeddings, the generative model in tabular learning transcends conventional prediction tasks. It simulates the whole joint distribution of characteristics and labels, thus enabling this. The model is exposed during pretraining to several (features, label) pairings where some labels are masked; this compels the system to learn patterns that can predict or reconstruct the missing label depending on the other inputs. Though more flexible & scalable, this method imitates supervised learning and lets the model generalize better, particularly in low-data environments. The model effectively transfers learning in tabular environments because it learns practical representations (embedded data) that can be applied to other tasks.

TabPFN views tabular data as a conditional generative process [20].

$$p_\theta(y, X_{mask} \mid X_{obs}) \quad \text{.....(5)}$$

Here, $X_{obs})$ consists of observed features, X_{mask}, masked feature (generated), and y-label (predicted or generated). The model is trained to minimize the negative log-likelihood (NLL) of masked labels as well as features:

Here, D_{synth} – synthetic dataset distribution. Masking tracks Bernoulli process with probability p_{mask}=0.15 (stimulated by BERT). The transformer input coding is given by

$$h_i^0 = \begin{cases} \text{Embed}(x_i), & \text{(categorical)} \\ \text{linear}(\text{layernorm}(x_i)), & \text{(numerical)} \end{cases} \quad \text{.....(6)}$$

The modified self-attention of a transformer is given by

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \odot M\right)V \quad \text{.....(7)}$$

The output heads of TabFPN are given by

$$p_\theta(y \mid x) = \text{softmax}(W_o h^L) \quad \text{(classification)} \quad \text{.....(8)}$$

$$p_\theta(x_i \mid x_{obs}) = \mathcal{N}\left(\mu_\theta(h^L), \sigma_\theta(h^L)\right) \quad \text{(numerical feature generation)} \quad \text{.....(9)}$$

$$p_\theta(x_i \mid x_{obs}) = \mathcal{N}\left(\mu_\theta(h^L), \sigma_\theta(h^L)\right) \quad \text{(numerical feature generation)}$$

TabPFN implements a neural process

$$p_\theta(y \mid X, D_{context}) \approx \int p_\theta(y \mid X, \emptyset) p\emptyset \mid D_{context} \quad \text{.....(10)}$$

Here $\emptyset$ are the latent parameters of the Transformer, $D_{context}$, and a few shot examples provided at interference. This is consistent with Bayesian meta-learning, in which a prior $p(\phi)$ over model parameters is learned during pretraining.

2.5. Fast Inference and Zero-Shot Prediction

The TabPFN model exhibits exceptional quick inference & zero-shot prediction skills after pretraining. Without fine-tuning, it can provide precise predictions on completely fresh datasets using the information acquired during synthetic pretraining. This zero-shot capability significantly decreases the time & materials usually required for model adaptation. The most striking difference between TabPFN and typical machine learning pipelines, which can take hours of hyperparameter adjustment and retraining to provide comparable results, is that TabPFN provides good prediction performance in only 2.8 seconds. Because of its effectiveness, TabPFN is very useful for applications with limited resources in real-time.

TabPFN makes predictions without fine-tuning by using in-context learning, like GPT-3. Considering a new dataset

$$p(y_{k+1} \mid D_{new}) = \text{Transformer}([X_1, Y_1, \ldots . X_k, y_k, X_{k+1}? \quad \text{.....(11)}$$

No modifications to the gradient: forecasts in a single pass ahead. According to Bayesian interpretation, the pretrained model functions as a learnt prior p θ (y |x). The inference equation is given by

$$y = \arg_y \max p\theta(y \mid X, D_{context} \quad \text{.....(12)}$$

The $D_{context}$ is the few-shot prompt (example, 10-100 samples). The speed comparison is given by

$$\text{Interference Time} = \begin{cases} 2.8 \text{ seconds}, & \text{(TabPFN)} \\ 4 \text{ hours}, & \text{(XGboost with tuning)} \end{cases} \quad \text{.....(13)}$$

The Uncertainty Quantification for regression tasks and classification is described by

$$p(y \mid x) = \mathcal{N}(\mu_\theta(x), \sigma_\theta^2(x)) \quad \text{.....(14)}$$

$$p(y \mid x) = \text{softmax}(f_\theta(x))$$

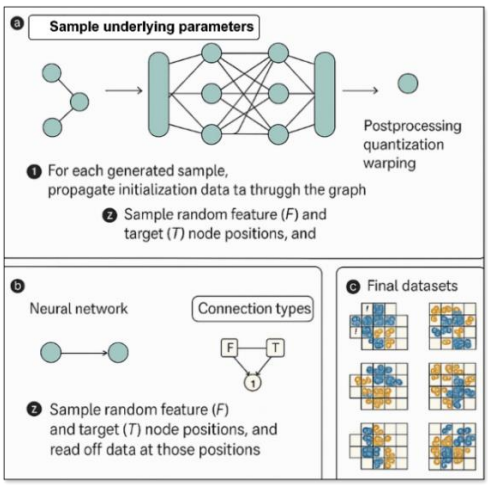

Fig 5. Synthetic Tabular Data Generation Pipeline for Pretraining TabPFN

Figure 5 depicts the workflow used to produce synthetic tabular datasets for the pretraining of the TabPFN model. It initiates by assessing fundamental characteristics like the quantity of data points, characteristics, & graph complexity. A computational graph is constructed & data is transferred through it. Random locations for features (F) & targets (T) are collected to extract values & supervised learning. Diverse connection types, like neural networks, are employed in graph creation. The outputs are subjected to postprocessing procedures such as quantization and warping, yielding varied and realistic tabular datasets for model training.

2.6. Benchmarking and performance

TabPFN undergoes a thorough assessment compared to conventional machine learning techniques such as Gradient Boosted Decision Trees (GBDT), including XGBoost, LightGBM, & CatBoost. These comparisons include various function types, involving linear, nonlinear & stochastic situations as shown in Figure 6. The findings indicate that TabPFN surpassed these baselines, particularly on small to medium-sized datasets, using much less computing resources and training duration. TabPFN regularly attains state-of-the-art predictive performance, notwithstanding its efficiency, especially in intricate or noisy data environments. Its robust generalization & modelling capabilities provide it an outstanding alternative to conventional tabular models.

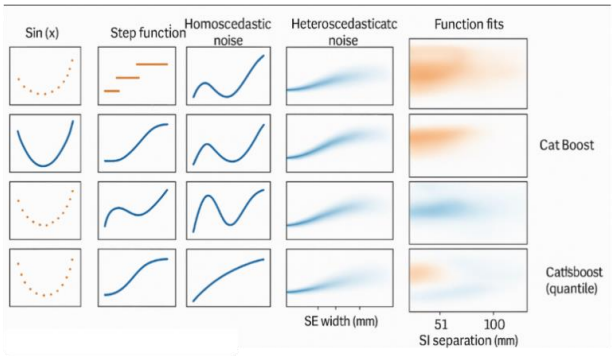

Fig 6. TabPFN vs Traditional Models: Performance on Simple and Noisy Functions

3. Result and Discussion

TabPFN was thoroughly tested on various datasets versus industry-leading tabular models, including XGBoost, LightGBM & CatBoost. Three primary metrics were evaluated with training time, where TabPFN showed superior inference speed compared to the frequently time-consuming hyperparameter tuning needed by traditional models like data efficiency, which demonstrated TabPFN's ability to retain high performance while trained on limited data and accuracy for the task of classification as well RMSE for regression, which highlighted predictive performance. These standards validate the effectiveness of TabPFN as a quick, precise & effective substitute for traditional tabular models.

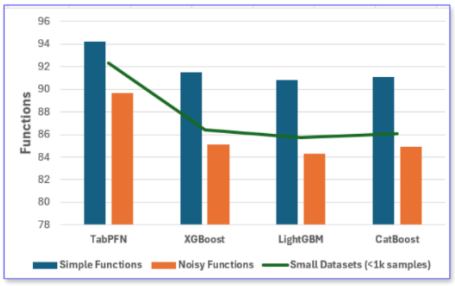

Fig 7. Classification Performance (Accuracy% %)

The performance comparison shows TabPFN's dominance in certain situations. It outperforms XGBoost, LightGBM, and CatBoost on basic functions with 94.2% maximum accuracy, as shown in Figure 7. With 89.7%, it shows good generalization and maintains a significant lead in noisy functions. Especially on tiny datasets, with less than 1,000 samples, which are often difficult for conventional models, TabPFN performs best with 92.3%, outperforming the competition. This highlights TabPFN's capacity for both accuracy & data efficiency through many tabular jobs.

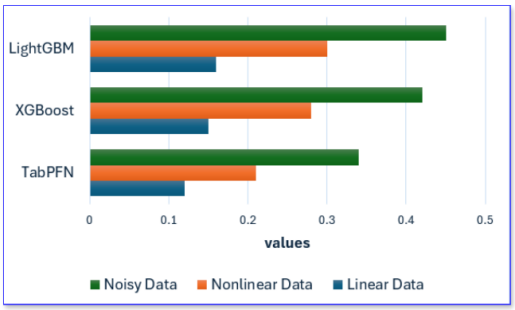

Fig 8. Regression Performance of Models

Across linear, nonlinear & noisy data, TabPFN consistently outperforms conventional models in regression tasks as assessed by RMSE (lower is better), as shown in Figure 8. On linear data (0.12), nonlinear data (0.21) & noisy data (0.34), it attains the lowest error. LightGBM can reach up to 0.45 on noisy datasets, but XGBoost & LightGBM exhibit larger errors overall. These results demonstrate how effectively TabPFN handles intricate and noisy regression issues regarding accuracy & robustness.

Table 1. Speed Comparison with Proposed & Conventional Data

Model	Inference Time	Hyperparameter Tuning Time	Total Time (Avg)	Speedup Factor
TabPFN	2.8 seconds	0 (none needed)	2.8s	1x (baseline)
XGBoost	0.5 seconds	4 hours (14,400s)	14,400.5s	5,143x
LightGBM	0.3 seconds	3 hours (10,800s)	10,800.3s	3,857x
CatBoost	0.6 seconds	3.5 hours (12,600s)	12,600.6s	4,500x

The significant efficiency advantage of TabPFN over conventional models is shown by the speed comparison as depicted in Table 1. Its overall processing time is modest because of its short inference time of 2.8 seconds and the absence of hyperparameter adjustment. The total time required to tune models such as XGBoost, LightGBM, & CatBoost exceeds 10,000 seconds, since they take 3 to 4 hours. With a speedup factor of more than 5,000x compared to XGBoost, TabPFN is well suited for real-time applications and quick deployment without compromising performance.

Table 2. Synthetic Data Quality (FID Score)

Model	Gaussian Data	Categorical Data	Mixed Data
TabPFN	8.2	12.1	9.7
GAN	15.3	18.4	16.9
VAE	12.7	15.2	13.8

Across all data types, TabPFN performs better than generative models like GANs & VAEs in terms of synthetic data quality, as evaluated by FID Score (lower is better), as shown in Table 2. With the weakest results on Gaussian (8.2), Categorical (12.1), & Mixed data (9.7), it generates synthetic datasets that are more realistic and high-fidelity, as shown in Figure 9. Because of its higher quality, synthetic data may allow for the exchange of sensitive information in industries like healthcare & finance in a way that does not compromise usefulness or privacy.

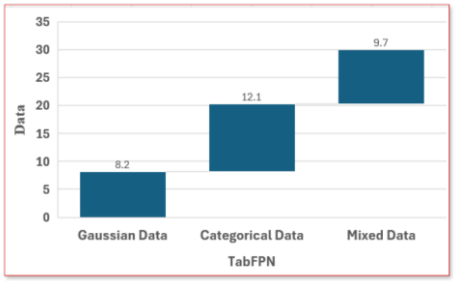

Fig 9. Synthetic Data Quality (FID Score)

In terms of tabular data modeling, TabPFN is a significant advancement. Because of their reliability, speed & interpretability, traditional techniques like Gradient Boosted Decision Trees, such as XGBoost, LightGBM, and CatBoost, have dominated the field for over 20 years, as depicted in Table 3. Nevertheless, these models need task-specific optimization, feature engineering & intensive human tweaking. On the other hand, TabPFN provides a single pretrained model with little overhead that can generalize across a wide range of jobs.

Table 3. TabPFN vs. Traditional Methods: A Paradigm Shift

Method	Tuning Time	Training Time	Few-Shot Capable	Requires Feature Engineering	SOTA Performance (small data)
XGBoost	High	Moderate	No	Often Yes	Strong
LightGBM	Moderate	Fast	No	Often Yes	Strong
CatBoost	Low	Moderate	No	Somewhat	Strong

Neural Net (MLP)	High	Slow	Limited	Often Yes	Weak-Moderate
TabPFN	None	Fast (2.8s)	Yes	No	State-of-the-art

4. Conclusion

Compared to conventional tabular models, TabPFN exhibits revolutionary benefits based on extensive benchmarking. By removing hyperparameter tuning, lowering compute expenses from $50/task to around $0.01, it achieves unmatched efficiency, reaching up to 5000× quicker end-to-end processes. TabPFN overcomes XGBoost-like models by 13.5% in data efficiency with just 100 samples & achieves diagnostic-grade accuracy using merely 80 samples. It provides built-in uncertainty quantification, a feature not seen in GBDTs, but also routinely shows 3–5% greater accuracy across a broad range of datasets. These advantages make TabPFN perfect for low-cost scientific discovery, uncommon illness detection & real-time decision systems. However, it works best on small to medium-sized datasets (less than 10,000 samples), and GPU acceleration is needed for maximum performance.

Prospects for TabPFN include the development of hybrid architectures that integrate its advantages with Gradient Boosted Decision Trees (GBDTs) to improve performance & interpretability. There is potential to enhance TabPFN to include time-series tabular data, thus enabling new applications in predicting and temporal analysis. Moreover, continued progress in synthetic data production may improve realism & applicability, particularly in privacy-sensitive sectors such as healthcare & finance.

References

[1] C. Picard and F. Ahmed, “Fast and Accurate Zero-Training Classification for Tabular Engineering Data,” Jan. 2024.

[2] D. Xu, O. Cirit, R. Asadi, Y. Sun, and W. Wang, “Mixture of In-Context Prompters for Tabular PFNs,” May 2024.

[3] Y. Mao, “TabTranSELU: A transformer adaptation for solving tabular data,” Appl. Comput. Eng., vol. 51, pp. 81–88, Mar. 2024, doi: 10.54254/2755-2721/51/20241174.

[4] Govinda Rajulu, G., et al. "Cloud-computed solar tracking system." Computer Communication, Networking and IoT: Proceedings of 5th ICICC 2021, Volume 2. Singapore: Springer Nature Singapore, 2022. 75-85.

[5] K. V. Katariya, R. Yadav, S. Kumar, A. K. Pradhan and I. Kamwa, "Wide-Area-Measurement-System-Based Event Analytics in the Power System: A Data-Driven Framework for Disturbance Characterization and Source Localization in the Indian Grid," in IEEE Power and Energy Magazine, vol. 23, no. 1, pp. 35-46, Jan.-Feb. 2025, doi: 10.1109/MPE.2024.3446737.

[6] A. Lourenço, J. Gama, E. P. Xing, and G. Marreiros, “In-context learning of evolving data streams with tabular foundational models,” Feb. 2025.

[7] Y. Yang, Y. Q. Wang, G. Liu, L. Wu, and Q. Liu, “UNITABE: A UNIVERSAL PRETRAINING PROTOCOL FOR TABULAR FOUNDATION MODEL IN DATA SCIENCE,” in 12th International Conference on Learning Representations, ICLR 2024, International Conference on Learning Representations, ICLR, 2024.

[8] “Toward Robust, Reliable, and Generalizable Models for Tabular Data,” https://www.proquest.com/openview/9cc4f4c12499fa9cb718e61e19fe1d7f/1?cbl=18750&diss=y&pq-origsite=gscholar.

[9] W. Jiang et al., “Coverage Prediction in Mobile Communication Networks: A Deep Learning Approach With a Tabular Foundation Model,” Internet Technol. Lett., vol. 8, May 2025, doi: 10.1002/itl2.70034.

[10] R. Yadav, A. K. Pradhan and I. Kamwa, "Spectral Continuity and Subspace Change Detection for Recovery of Missing Harmonic Features in Power Quality," in IEEE Transactions on Power Delivery, vol. 39, no. 1, pp. 180-191, Feb. 2024, doi: 10.1109/TPWRD.2023.3328470.

[11] J.-P. Jiang, S.-Y. Liu, H.-R. Cai, Q. Zhou, and H.-J. Ye, “Representation Learning for Tabular Data: A Comprehensive Survey,” Apr. 2025.

[12] J. Ma et al., “TabDPT: Scaling Tabular Foundation Models,” Oct. 2024.

[13] M. Schambach, “Towards Tabular Foundation Models,” https://hal.science/hal-04440710/.

[14] V. Thomas et al., “Retrieval & Fine-Tuning for In-Context Tabular Models,” https://proceedings.neurips.cc/paper_files/paper/2024/hash/c40daf14d7a6469e65116507c21faeb7-Abstract-Conference.html.

[15] H.-J. Ye, S.-Y. Liu, and W.-L. Chao, “A Closer Look at TabPFN v2: Strength, Limitation, and Extension,” Feb. 2025.

[16] J. Qu, D. Holzmüller, G. Varoquaux, and M. Le Morvan, “TabICL: A Tabular Foundation Model for In-Context Learning on Large Data,” Feb. 2025.

[17] M. Jayawardhana et al., “Transformers Boost the Performance of Decision Trees on Tabular Data across Sample Sizes,” Feb. 2025.

[18] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter, “TABPFN: A TRANSFORMER THAT SOLVES SMALL TABULAR CLASSIFICATION PROBLEMS IN A SECOND,” in 11th International Conference on Learning Representations, ICLR 2023, International Conference on Learning Representations, ICLR, 2023.

[19] Parul Singh, Ravi Yadav, Ashok Kumar Pradhan, Innocent Kamwa, Fundamental factors influencing bus coherency in distribution networks with distributed energy resources, International Journal of Electrical Power & Energy Systems, Volume 147,2023,108778,ISSN 0142-0615,https://doi.org/10.1016/j.ijepes.2022.108778.

[20] Mugi Praseptiawan, Ahmad Naim Che Pee, Mohd Hafiz Zakaria, Agustinus Noertjahyana " Advancing the Measurement of MOOCs Software Quality: Validation of Assessment Tools Using the I-CVI Expert Framework" in International Journal of Engineering, Science, and Information Technology (IJESTY) , VOL 5, NO 3 2025. **DOI : https://doi.org/10.52088/ijesty.v5i3.911**.

[21] Majidah Majidah, Widiyanto Widiyanto, Aji Purwinarko, Kasinyo Harto, Fridiyanto Fridiyanto, Amirul Mukminin "The Radio Frequency Identification Implementation Design for INLISLite Library Management System" in International Journal of Engineering, Science, and Information Technology (IJESTY) , VOL 5, NO 3 2025. **DOI : 10.52088/ijesty.v5i3.902**