Drift-Resilient TabPFN: In-Context Learning Temporal Distribution Shifts on Tabular Data

David Schnurr^{*,1,2} **Kai Helli**^{*,1,3} **Noah Hollmann**^{1,4} **Samuel Müller**^{1} **Frank Hutter**^{5,1}
^{1} University of Freiburg, ^{2} ETH Zurich, ^{3} Technical University of Munich,
^{4} Charité University Medicine Berlin, ^{5} ELLIS Institute Tübingen, * Equal contribution.
Correspondence to `kai.helli@tum.de`

Abstract

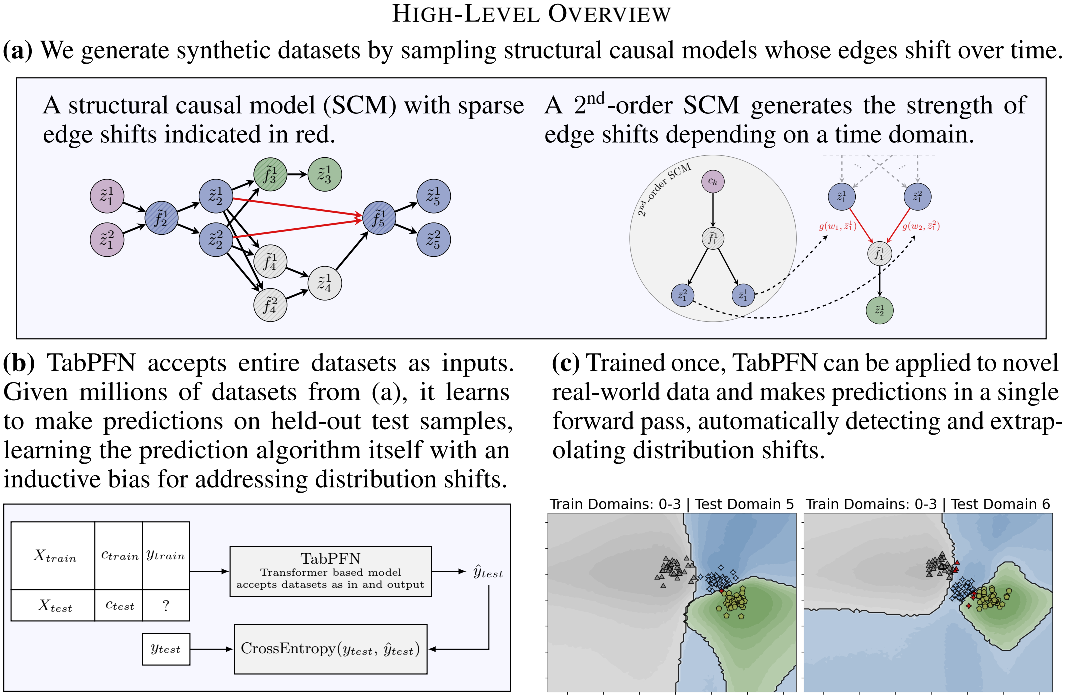
While most ML models expect independent and identically distributed data, this assumption is often violated in real-world scenarios due to distribution shifts, resulting in the degradation of machine learning model performance. Until now, no tabular method has consistently outperformed classical supervised learning, which ignores these shifts. To address temporal distribution shifts, we present Drift-Resilient TabPFN, a fresh approach based on In-Context Learning with a Prior-Data Fitted Network that learns the learning algorithm itself: it accepts the entire training dataset as input and makes predictions on the test set in a single forward pass. Specifically, it learns to approximate Bayesian inference on synthetic datasets drawn from a prior that specifies the model's inductive bias. This prior is based on structural causal models (SCM), which gradually shift over time. To model shifts of these causal models, we use a secondary SCM, that specifies changes in the primary model parameters. The resulting Drift-Resilient TabPFN can be applied to unseen data, runs in seconds on small to moderately-sized datasets and needs no hyperparameter tuning. Comprehensive evaluations across 18 synthetic and real-world datasets demonstrate large performance improvements over a wide range of baselines, such as XGB, CatBoost, TabPFN, and applicable methods featured in the Wild-Time benchmark. Compared to the strongest baselines, it improves accuracy from 0.688 to 0.744 and ROC AUC from 0.786 to 0.832 while maintaining stronger calibration. This approach could serve as significant groundwork for further research on out-of-distribution prediction.

1 Introduction

In traditional machine learning, the train and test data are assumed to be sampled from the same distribution [1]. However, this assumption of independent and identically distributed (i.i.d.) data is commonly violated in real-world scenarios due to distribution shifts, resulting in performance degradation of standard machine learning (ML) models over time [1, 2]. Research in the area of temporal domain generalization (Temporal DG) tries to address these shifts by developing methods that perform consistently across temporal domains and generalize beyond the training regimen, i.e. into the future. In fields such as healthcare, climate science, or finance, data is most often organized in a tabular format [3, 4]. Here shifts are driven by hidden variables such as policy or climate changes, equipment updates, seasonal changes, or activity cycles, limiting real-world model deployment [2].

Robustness to such distribution shifts stands as a prominent challenge in current ML research [5]. So far multiple approaches have been proposed to address temporal distribution shifts using neural networks (NNs) [5–7]. However, modeling distribution shifts in tabular data presents a two-fold challenge: (i) NNs have struggled to model and extrapolate distribution shifts to date [5, 8] (ii) approaches for modeling distribution shifts have mostly employed NNs, while tree-based

Table Representation Learning Workshop at NeurIPS 2024



HIGH-LEVEL OVERVIEW

(a) We generate synthetic datasets by sampling structural causal models whose edges shift over time.

A structural causal model (SCM) with sparse edge shifts indicated in red.

A 2nd-order SCM generates the strength of edge shifts depending on a time domain.

(b) TabPFN accepts entire datasets as inputs. Given millions of datasets from (a), it learns to make predictions on held-out test samples, learning the prediction algorithm itself with an inductive bias for addressing distribution shifts.

(c) Trained once, TabPFN can be applied to novel real-world data and makes predictions in a single forward pass, automatically detecting and extrapolating distribution shifts.

Train Domains: 0-3 | Test Domain 5
Train Domains: 0-3 | Test Domain 6

Figure 1: High-level overview of our method. We train a transformer that accepts entire datasets as input to learn the learning algorithm itself by training on millions of synthetic datasets once as part of algorithm development. The trained model can be applied to arbitrary real-world datasets. In (b), X, c, and y refer to features, time domain, and label, respectively. In (c), we show predictions on test domains 4 (left) and 5 (right), where we see a distribution shift. Drift-Resilient TabPFN accurately updates decision boundaries in this example.

methods have consistently outperformed NNs in handling tabular data [3, 9–11] - leaving a wide methodological gap in addressing this common real-world scenario.

We provide a fresh perspective on predicting given distribution shifts by leveraging in-context-learning (ICL) to learn the prediction algorithm itself - bypassing many challenges encountered in this setting. Our approach builds on the foundation of Prior-Data Fitted Networks (PFNs; 12) and TabPFN (13; see Section 3). PFNs leverage large-scale ML and ICL techniques to approximate Bayesian inference accurately for any prior that can be sampled from. They are trained on millions of synthetic datasets sampled from this prior. For each such dataset, a supervised learning task is constructed, and the model is asked to predict on held out test samples. Then, the PFN is able to apply the principles learned on this synthetic data to real-world datasets, effectively having learned a prediction algorithm.

This paper introduces *Drift-Resilient TabPFN*, an adaptation of the TabPFN framework tailored for tabular datasets exhibiting temporal distribution shifts. Our idea is as follows: Data distribution shifts can be modeled as gradual changes to the structural causal model (SCM; 14, 15) underlying the data. By including the assumption that underlying models change over time into the approximated prior, our models learn to estimate, adapt to, and extrapolate these model changes.

2 Background

Consider $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{C}$ as the sample spaces for features, labels, and domain indices, respectively, with specific instances represented by $\boldsymbol{x}$, y, and $\boldsymbol{c}$. The corresponding random variables are X, Y, and C.

A dataset is a collection of n tuples, $\mathcal{D} := \{(\boldsymbol{x}_i, y_i, \hat{\boldsymbol{c}}_k)\}_{i=1}^{n}$, each drawn from a conditional distribution $\mathbb{P}(X, Y \mid C = \boldsymbol{c}_k)$ over $\mathcal{X} \times \mathcal{Y}$. Here, $\boldsymbol{c}_k \in \mathcal{C}$ serves as a domain index conditioning the sample distribution. Since the true temporal domain is often unknown in real-world data, it is approximated as $\hat{\boldsymbol{c}}_k$ in the dataset. To isolate samples from a specific domain $\hat{\boldsymbol{c}}_k$, we define the sub-dataset $\mathcal{D}_{\hat{\boldsymbol{c}}_k} := \{(\boldsymbol{x}, y, \hat{\boldsymbol{c}}) \in \mathcal{D} \mid \hat{\boldsymbol{c}} = \hat{\boldsymbol{c}}_k\}$. This allows for a more nuanced analysis of datasets with domain shifts.

Domain Generalization (DG) aims to train a model that generalizes from source domains $\mathcal{C}^{\text{train}}$ to target domains $\mathcal{C}^{\text{test}}$ without accessing target domain samples. The objective is to learn a mapping function $f : \mathcal{X} \times \mathcal{C} \to \mathcal{Y}$ that minimizes expected loss on unseen target domains. Temporal Domain Generalization (Temporal DG), a special case of DG, has a one-dimensional domain index set $\mathcal{C}$ that follows a total ordering, $c_1 \leq c_2 \leq \ldots$. The training set is limited to source domains that precede target domains, $\mathcal{C}^{\text{train}} = \{c_1, c_2, \ldots, c_t\}$, and the objective is to learn a predictive model f that generalizes to future, unseen domains $\mathcal{C}^{\text{test}} = \{c_{t+1}, c_{t+2}, \ldots, c_n\}$. The indices of all training domains $\mathcal{C}^{\text{train}}$ and the index of the current testing domain $c_k \in \mathcal{C}^{\text{test}}$ are provided to the model.

3 Methodology

Our approach is built on PFNs, which use ICL to learn the learning algorithm itself. This approach also has a theoretical foundation as described by Müller et al. [12]: It can be viewed as approximating Bayesian prediction for a prior defined by the synthetic datasets. The trained PFN will approximate the posterior predictive distribution (PPD) and thus return a Bayesian prediction for the specified distribution over artificial datasets used during PFN training.

Hollmann et al. [13] introduce a prior based on Structural Causal Models (SCMs; 14; 15) to model complex feature dependencies and potential causal mechanisms underlying tabular data. To sample one dataset, this prior samples an SCM, which is then used to sample the examples in the dataset. In this approach, each causal representation of a sampled SCM is converted into a functional representation to enable forward computation and dataset sampling. We extend TabPFN's prior to model distribution shifts, allowing the model to expand its posterior predictive distribution (PPD) calculations to incorporate temporal domain information. We propose modeling distribution shifts via shifting edges of the SCM over our temporal domain. Furthermore, we introduce a 2^{nd}*-order SCM* to model these shifts. This 2^{nd}-order SCM is itself an SCM with feature nodes specifying the magnitude of edge shifts in the base SCM's functional graph.

To do this, we sample the temporal domains $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ and for each domain c_k, we sample the number of samples n_{c_k} it contains. An illustration showing exemplary domain distributions across four datasets can be found in Figure 5 in Appendix A.5.

We then select a sparse subset of relationships in the causal representation of the SCM to undergo temporal shifts based on the evidence that sparse shifts allow for causal reasoning [1]. For these selected edges, the 2^{nd}-order SCM is used to sample shift parameters that govern the corresponding edges in the functional representation. It takes temporal domains $\mathcal{C}$ as input and, through a single forward pass on the corresponding functional representation, produces dynamic edge shifts for each edge weight $w_{i,j}$ in the original SCM that corresponds to a causal relationship that should be shifted. This design allows for Bayesian reasoning over the edge shifts and enables the 2^{nd}-order SCM to generate complex, often correlated shifts over time. For better illustration, we have visualized this approach in Figure 2 and a selection of the functions generated by a 2^{nd}-order SCM in Figure 6. A high-level outline of the sampling procedure is detailed in Algorithm 1.

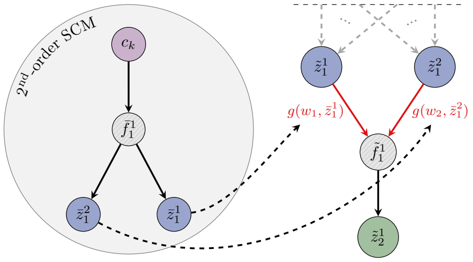


Figure 2: Diagram illustrating the integration of a 2^{nd}-order SCM for adaptive edge shifting across evolving temporal domains. On the right, the primary network $\tilde{\mathcal{G}}$ generates data samples over multiple time domains, with red arrows indicating shifted edges. On the left, the 2^{nd}-order SCM - an auxiliary network $\tilde{\mathcal{H}}$ - takes an input domain $c_k \in \mathcal{C}$ and outputs parameters to adaptively shift each edge weight w_i in the base network.

4 Experiments

We evaluate the performance using in-distribution (ID) and out-of-distribution (OOD) splits across multiple datasets, employing state-of-the-art tabular prediction methods, including TabPFN variants. Detailed information about the evaluation strategy, metrics, datasets, baselines, and TabPFN setup is provided in Appendix A.2.

Quantitative Evaluation Our method demonstrates superior predictive performance in all metrics for OOD data across 18 test datasets, as detailed in Table 3. Compared to the strongest baseline, it improves accuracy from 0.688 to 0.744, F1 from 0.62 to 0.689, and ROC from 0.786 to 0.832. Furthermore, Drift-Resilient TabPFN shows much stronger calibration on OOD samples, improving ECE from 0.119 to 0.091. While baselines are often overconfident on OOD data, Drift-Resilient TabPFN is able to predict uncertainty accurately. Since our method focuses on enhancing OOD robustness rather than optimizing ID tasks, we find lower predictive performance on ID tasks. While performance gains are observed on both, real-world and synthetic data, we observe stronger improvements on synthetic datasets. This can be partly attributed to the, on average, stronger distribution shifts between ID and OOD data in our synthetic benchmark. Furthermore, real-world datasets often show multifaceted and complex shifts that are much more difficult to extrapolate into the future. A split evaluation of synthetic and real-world can be found in Appendix A.9.

Qualitative Analysis. Next, we take an in-depth look at the predictions made by our method. Figure 3 illustrates the decision boundaries of our method and TabPFN$_{\text{base}}$ on the synthetic Intersecting Blobs dataset. In this evaluation, we restrict the training domains to $\mathcal{C}^{\text{train}} = \{0, 1, 2, 3\}$ and aim to predict samples in test domains $\mathcal{C}^{\text{test}} = \{4, 5, 6\}$ without adding additional data to the training set. This setup requires the model to extrapolate the temporal shifts into the future based solely on existing training data. In this setting, our model accurately extrapolates decision boundaries to future domains, while TabPFN$_{\text{base}}$ tends to retain its initial boundary. Our analysis reveals two key attributes of our model: (i) The model decreases prediction certainty over time, improving calibration. (ii) Our model adjusts the decision boundary dynamically, boosting accuracy.

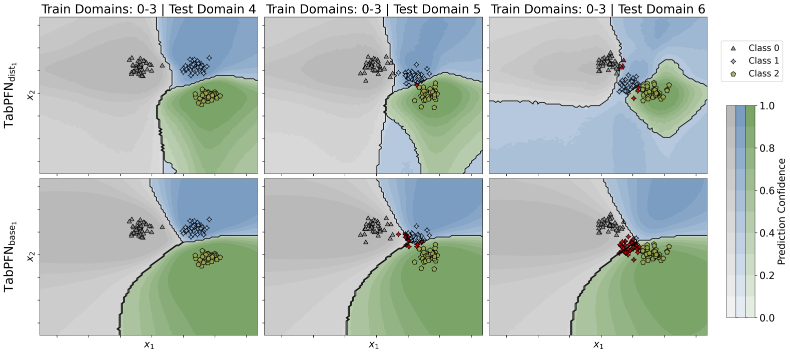


Figure 3: This figure displays the predictive behavior of TabPFN$_{\text{dist}}$ in the top row and TabPFN$_{\text{base}}$ in the bottom row on the Intersecting Blobs dataset. It illustrates how each model adapts to unseen test domains when trained on domains $\mathcal{C}^{\text{train}} = \{0, 1, 2, 3\}$. The baseline is given the domain indices as a feature in train and test. The coloring indicates the probability of the most likely class at each point. Incorrectly classified samples are highlighted in red.

5 Conclusions

In this work, we presented a Bayesian approach to address the issue of temporal domain generalization in tabular data. Specifically, we focused on enhancing TabPFN to improve its robustness to temporal distribution shifts. Within this framework, we introduced a novel approach that changes the causal relationships in the SCM prior over time, thereby enabling TabPFN to inherently adapt to these shifts. Our method outperforms all baselines on the evaluated datasets and demonstrates notable improvements both qualitatively and quantitatively, particularly on synthetic OOD datasets. Furthermore, it requires no hyperparameter tuning, is not limited to particular types of distribution shifts, and takes only 10.9s for training and prediction combined.

6 Acknowledgments

Frank Hutter acknowledges the financial support of the Hector Foundation. This research was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant number 417962828.

We acknowledge funding by the European Union (via ERC Consolidator Grant DeepLearning 2.0, grant no. 101045765). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Funded by
the European Union

References

[1] Ronan Perry, Julius Von Kügelgen, and Bernhard Schölkopf. Causal discovery in heterogeneous environments under the sparse mechanism shift hypothesis. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022. URL `https://openreview.net/forum?id=kFRCvpubDJo`.

[2] Daniel Vela, Andrew Sharp, Richard Zhang, Trang Nguyen, An Hoang, and Oleg S. Pianykh. Temporal quality degradation in AI models. *Scientific Reports*, 12(1):11654, July 2022. ISSN 2045-2322. doi:10.1038/s41598-022-15245-z.

[3] 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*, 2022.

[4] Boris van Breugel and Mihaela van der Schaar. Why tabular foundation models should be a research priority. *arXiv preprint arXiv:2405.01147*, 2024.

[5] Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei Koh, and Chelsea Finn. Wild-time: A benchmark of in-the-wild distribution shift over time. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022. URL `https://openreview.net/forum?id=F9ENmZABB0`.

[6] Guangji Bai, Chen Ling, and Liang Zhao. Temporal domain generalization with drift-aware dynamic neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR'23)*, 2023. URL `https://openreview.net/forum?id=sW0sRj4nT1n`. Published online: `iclr.cc`.

[7] Anshul Nasery, Soumyadeep Thakur, Vihari Piratla, Abir De, and Sunita Sarawagi. Training for the future: A simple gradient interpolation loss to generalize along time. In M. Ranzato, A. Beygelzimer, K. Nguyen, P. Liang, J. Vaughan, and Y. Dauphin, editors, *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'21)*. Curran Associates, 2021. URL `https://openreview.net/forum?id=U7SBcmRf65`.

[8] Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with tableshift. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34: 18932–18943, 2021.

[10] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[11] Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022. URL `https://openreview.net/forum?id=Fp7__phQszn`.

[12] S. Müller, N. Hollmann, S. Arango, J. Grabocka, and F. Hutter. Transformers can do bayesian inference. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2022. URL `https://openreview.net/forum?id=KSugKcbNf9`. Published online: `iclr.cc`.

[13] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *Proceedings of the International Conference on Learning Representations (ICLR'23)*, 2023. URL `https://openreview.net/forum?id=cp5PvcI6w8_`. Published online: `iclr.cc`.

[14] Judea Pearl. *Causality*. Cambridge University Press, 2 edition, 2009.

[15] J. Peters, D. Janzing, and B. Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[16] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*. Omnipress, 2013. URL `https://proceedings.mlr.press/v28/muandet13.html`.

[17] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'18)*. Curran Associates, 2018.

[18] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5715–5725, 2017.

[19] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2020.

[20] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *Proceedings of the International Conference on Learning Representations (ICLR'20)*, 2020. Published online: `iclr.cc`.

[21] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proceedings of the International Conference on Learning Representations (ICLR'18)*, 2018. Published online: `iclr.cc`.

[22] Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation, 2022.

[23] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In Gang Hua and Hervé Jégou, editors, *Computer Vision – ECCV 2016 Workshops*, pages 443–450, Cham, 2016. Springer International Publishing. doi:10.1007/978-3-319-49409-8_35.

[24] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030, jan 2016. ISSN 1532-4435.

[25] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning (ICML'21)*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021. URL `https://proceedings.mlr.press/v139/krueger21a.html`.

[26] Cian Eastwood, Alexander Robey, Shashank Singh, Julius von Kügelgen, Hamed Hassani, George J. Pappas, and Bernhard Schölkopf. Probable domain generalization via quantile risk minimization. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*, 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/6f11132f6ecbbcafafdf6decfc98f7be-Paper-Conference.pdf`.

[27] Andrey Malinin, Neil Band, Alexander Ganshin, German Chesnokov, Yarin Gal, Mark J. F. Gales, Alexey Noskov, Andrey Ploskonosov, Liudmila Prokhorenkova, Ivan Provilkov, Vatsal Raina, Vyas Raina, Denis Roginskiy, Mariya Shmatova, Panos Tigar, and Boris Yangel. Shifts: A dataset of real distributional shift across multiple large-scale tasks. *arXiv preprint arXiv:2107.07455*, 2021.

[28] Andrey Malinin, Andreas Athanasopoulos, Muhamed Barakovic, Meritxell Bach Cuadra, Mark J. F. Gales, Cristina Granziera, Mara Graziani, Nikolay Kartashev, Konstantinos Kyriakopoulos, Po-Jui Lu, Nataliia Molchanova, Antonis Nikitakis, Vatsal Raina, Francesco La Rosa, Eli Sivena, Vasileios Tsarsitalidis, Efi Tsompopoulou, and Elena Volf. Shifts 2.0: Extending the dataset of real distributional shifts, 2022. URL `https://arxiv.org/abs/2206.15407`.

[29] Jiashuo Liu, Tianyu Wang, Peng Cui, and Hongseok Namkoong. On the need for a language describing distribution shifts: Illustrations on tabular datasets. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[30] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. In M. Ranzato, A. Beygelzimer, K. Nguyen, P. Liang, J. Vaughan, and Y. Dauphin, editors, *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'21)*. Curran Associates, 2021.

[31] Sergey Kolesnikov. Wild-tab: A benchmark for out-of-distribution generalization in tabular regression, 2023. URL `https://arxiv.org/abs/2312.01792`.

[32] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *Proceedings of the International Conference on Learning Representations (ICLR'21)*, 2021. URL `https://openreview.net/forum?id=lQdXeXDoWtI`. Published online: `iclr.cc`.

[33] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8052–8072, 2023. doi:10.1109/TKDE.2022.3178128.

[34] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In A. Globerson and R. Silva, editors, *Proceedings of The 34th Uncertainty in Artificial Intelligence Conference (UAI'18)*. AUAI Press, 2018.

[35] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. Time2vec: Learning a vector representation of time, 2020. URL `https://openreview.net/forum?id=rklklCVYvB`.

[36] Junwei Ma, Valentin Thomas, Guangwei Yu, and Anthony Caterini. In-context data distillation with tabpfn, 2024. URL `https://arxiv.org/abs/2402.06971`.

[37] Benjamin Feuer, Robin Tibor Schirrmeister, Valeriia Cherepanova, Chinmay Hegde, Frank Hutter, Micah Goldblum, Niv Cohen, and Colin White. Tunetables: Context optimization for scalable prior-data fitted networks. *arXiv preprint arXiv:2402.11137*, 2024.

[38] Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V Naidu, and Colin White. Forecastpfn: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, 2023.

[39] L. Prokhorenkova, G. Gusev, A. Vorobev, A. Dorogush, and A. Gulin. CatBoost: unbiased boosting with categorical features. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'18)*. Curran Associates, 2018.

[40] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In B. Krishnapuram, M. Shah, A. Smola, C. Aggarwal, D. Shen, and R. Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16)*, pages 785–794. ACM Press, 2016.

[41] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Proceedings of the 30th International Conference on Advances in Neural Information Processing Systems (NeurIPS'17)*. Curran Associates, 2017.

[42] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[43] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In D. Precup and Y. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML'17)*, volume 70. Proceedings of Machine Learning Research, 2017.

[44] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *Proceedings of the International Conference on Learning Representations (ICLR'19)*, 2019. Published online: `iclr.cc`.

[45] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In H. Daume III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*, volume 98. Proceedings of Machine Learning Research, 2020.

[46] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In H. Larochelle, M. Ranzato, R. Hadsell, M.-F. Balcan, and H. Lin, editors, *Proceedings of the 33rd International Conference on Advances in Neural Information Processing Systems (NeurIPS'20)*. Curran Associates, 2020.

[47] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. URL `http://www.jstor.org/stable/3001968`.

[48] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.

[49] Salvador Garcia and Francisco Herrera. An extension on" statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of machine learning research*, 9 (12), 2008.

[50] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[51] Bendi Ramana and N. Venkateswarlu. ILPD (Indian Liver Patient Dataset). UCI Machine Learning Repository, 2012.

[52] Oguz Akbilgic. Istanbul Stock Exchange. UCI Machine Learning Repository, 2013.

[53] Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore. Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records. *BioMed Research International*, 2014:781670, April 2014. ISSN 2314-6133. doi:10.1155/2014/781670. URL `https://doi.org/10.1155/2014/781670`. Publisher: Hindawi Publishing Corporation.

[54] Albert Bifet and Elena Ikonomovska. Data Expo competition. OpenML, 2009. URL `https://www.openml.org/search?type=data&sort=runs&id=1169&status=active`.

[55] Jack Smith, J. Everhart, W. Dickson, W. Knowler, and Richard Johannes. Using the adap learning algorithm to forcast the onset of diabetes mellitus. *Proceedings - Annual Symposium on Computer Applications in Medical Care*, 10, 11 1988.

[56] M. M. Faniqul Islam, Rahatara Ferdousi, Sadikur Rahman, and Humayra Yasmin Bushra. Likelihood prediction of diabetes at early stage using data mining techniques. In Mousumi Gupta, Debanjan Konar, Siddhartha Bhattacharyya, and Sambhunath Biswas, editors, *Computer Vision and Machine Intelligence in Medical Image Analysis*, pages 113–125, Singapore, 2020. Springer Singapore. ISBN 978-981-13-8798-2. doi:10.1007/978-981-13-8798-2_12.

[57] Luis Candanedo. Occupancy Detection. UCI Machine Learning Repository, 2016.

[58] Ricardo Ferreira, Andrea Martiniano, and Renato Sassi. Behavior of the urban traffic of the city of Sao Paulo in Brazil. UCI Machine Learning Repository, 2018.

[59] Jacob Montiel, Jesse Read, Albert Bifet, and Talel Abdessalem. Scikit-multiflow: A multi-output streaming framework. *Journal of Machine Learning Research*, 19(72):1–5, 2018. URL `http://jmlr.org/papers/v19/18-251.html`.

[60] Angela Dispenzieri, Jerry Katzmann, Robert Kyle, Dirk Larson, Terry Therneau, Colin Colby, Raynell Clark, Graham Mead, Shaji Kumar, L Melton, and S Rajkumar. Use of nonclonal serum immunoglobulin free light chains to predict overall survival in the general population. *Mayo Clinic proceedings. Mayo Clinic*, 87:517–23, 06 2012. doi:10.1016/j.mayocp.2012.03.009.

[61] Robert Kyle, Terry Therneau, S Rajkumar, Dirk Larson, Matthew Plevak, Janice Offord, Angela Dispenzieri, Jerry Katzmann, and L Melton. Prevalence of monoclonal gammopathy of undetermined significance. *The New England journal of medicine*, 354:1362–9, 04 2006. doi:10.1056/NEJMoa054494.

[62] Michael Harries. *Splice-2 comparative evaluation: Electricity Pricing*. University of New South Wales, School of Computer Science and Engineering [Sydney], 1999. URL `http://nla.gov.au/nla.arc-32869`.

[63] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In Ana L. C. Bazzan and Sofiane Labidi, editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 286–295, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-28645-5. doi:10.1007/978-3-540-28645-5_29.

[64] Andrea Martiniano and Ricardo Ferreira. Absenteeism at work. UCI Machine Learning Repository, 2018.

[65] Andras Janosi, William Steinbrunn, Matthias Pfisterer, and Robert Detrano. Heart Disease. UCI Machine Learning Repository, 1988.

[66] Daniel Stolfi. Parking Birmingham. UCI Machine Learning Repository, 2019.

[67] Dean De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19, 11 2011. doi:10.1080/10691898.2011.11889627.

[68] Indrė Žliobaitė. Combining similarity in time and space for training set formation under concept drift. *Intelligent Data Analysis*, 15(4):589–611, 2011.

A Appendix

A.1 Related Work

While DG has drawn increasing attention in the research community [16–26], its temporal variant remains under-explored. In this section, we review existing DG benchmarks on tabular data, DG methods, the temporal DG benchmark Wild-Time [5], specialized temporal DG methods, as well as relevant studies of TabPFN.

Tabular Distribution Shift Benchmarks. Several benchmarks have been introduced to assess methods for tabular data under distribution shifts. (1) **Shifts** and **Shifts 2.0** [27, 28] are uncertainty focused benchmarks. **Shifts 2.0** includes five tasks, two of which involve tabular data subject to temporal and spatiotemporal shifts. (2) **WhyShift** [29], focusing on spatio-temporal shifts, offers five real-world tabular datasets, including the ACS dataset [30], which we also use in a subsampled form. Evaluating 22 methods like Gradient Boosted Decision Trees (GBDTs), MLPs, and robustness techniques, WhyShift finds that robustness methods do not consistently enhance out-of-distribution (OOD) performance. They also find that while GBDTs perform better, the gap between in-distribution (ID) and OOD performance persists, likely due to GBDTs being better fitted to the ID distribution. (3) **TableShift** [8] includes 15 tabular binary classification tasks, with 10 relevant to DG. Their work evaluates 19 model types, including several DG methods, finding that methods tailored for distribution shifts do not consistently outperform GBDTs. While generalization gaps can be slightly reduced, each robustness method comes at the cost of ID performance. (4) **Wild-Tab** [31] focuses on domain generalization within tabular regression using three large datasets. Their study compares 10 generalization techniques against standard Empirical Risk Minimization (ERM) applied to MLPs. Similar to previous work [32], they find that ERM was not consistently outperformed by specialized DG methods, and notably, no advantage of GBDTs over ERM on MLPs was observed in their datasets.

Unlike these benchmarks, which focus on large-scale datasets, our work addresses the overlooked challenges of small-scale temporal distribution shift datasets. However, their insights on generalization and robustness methods closely align with our findings, providing valuable context for our approach.

DG Methods. Several techniques have been proposed to improve robustness to distribution shifts in DG [33]. Key approaches include domain-invariant learning methods, such as **Deep CORAL** [23], **IRM** [19], and **DANN** [24], which aim to learn representations that generalize across domains. Data augmentation strategies, like **Mixup** [21] and **LISA** [22], contribute to generalization by generating synthetic data variations. Additionally, robust optimization techniques, including **VRex** [25], **GroupDRO** [20], **EQRM** [26], and **SWA** [34], aim to improve performance under distributional shifts by optimizing for worst-case scenarios or incorporating model uncertainty.

In relation to our work, data augmentation strategies are conceptually similar, as we also teach the model DG rather than adding invariances to the architecture. In contrast to augmentation strategies, though, we learn to generalize using completely artificial data on the meta level rather than through manipulation of the target dataset. The other methods focus on designing models specifically for DG, while our approach completely relies on the model to learn to handle distribution shifts.

Wild-Time Benchmark. Wild-Time [5] is a benchmark of five datasets designed to study the real-world effects of temporal distribution shifts - an area largely overlooked by previous benchmarks. While Wild-Time primarily uses non-tabular data, it evaluates a wide array of techniques on its tabular dataset, including classical supervised learning (ERM), fine-tuning, and several previously mentioned general DG methods adapted to temporal distribution shifts. Despite the diversity of methods evaluated, Wild-Time reveals a significant performance gap between ID and OOD data, with none of the 13 tested methods consistently outperforming the standard ERM approach.

In our evaluation, we employ their evaluation strategy *Eval-Fix* and also benchmark our approach against the methods they considered. A comprehensive overview of these methods is provided in Appendix A.7.

Recent Temporal DG methods. Recent specialized temporal DG methods include: (1) **DRAIN** [6], which employs a Bayesian framework alongside a recurrent neural network for predicting the dynamics of the model parameters across temporal domains. (2) **GI** [7], which explicitly incorporates the temporal domain as a feature, using a specialized Gradient Interpolation loss function, a timesensitive activation, and enhanced domain reasoning via Time2Vec preprocessing [35]. However, both DRAIN and GI are limited in their ability to extrapolate beyond the near future, leading to their exclusion from our main evaluation. Notably, even on the Rotated Two Moons Dataset - where DRAIN and GI have demonstrated their capabilities - our approach Drift-Resilient TabPFN, outperforms both. See Appendix A.9.8 for details.

Recent TabPFN Studies. To overcome the limitations of TabPFN in handling large samples and feature sets, typically found in DG benchmarks, several improvements have been proposed. Ma et al. [36] introduced a data distillation approach, where a distilled dataset serves as the model's context, optimized to maximize the training-data likelihood. Similarly, TuneTables [37], uses prompt-tuning to compress large datasets into a smaller context. Either of these methods could potentially be combined with Drift-Resilient TabPFN, as our modifications focus primarily on the pre-training phase, which remains unchanged in their approaches.

Another TabPFN variation, ForecastPFN [38], also introduces time dependence, but it does not consider any features. It only models simple time series data. Unlike our approach, which builds on TabPFN's SCM architecture for pre-training to handle a large set of features, ForecastPFN models synthetic time series using a single handcrafted function with sampled hyperparameters, simplifying the architecture while trading off diversity of the synthetic datasets.

A.2 Experimental Setup

Evaluation Strategy. We evaluate analogous to the Eval-Fix setting outlined in Wild-Time [5], measuring both, in-distribution (ID) and out-of-distribution (OOD) performance. Here, each dataset $\mathcal{D}$ is split into three subsets: $\mathcal{D}^{\text{train}}$, $\mathcal{D}^{\text{ID}}$, and $\mathcal{D}^{\text{OOD}}$. Splits are based on a randomly sampled temporal domain c_k that serves as the boundary between the train and test (OOD) portion. We only use such splits, where $\mathcal{D}^{\text{train}}$ comprises between 30% and 80% of the total domains and samples. To assess ID performance, we subsample 10% of the instances in each domain of $\mathcal{D}^{\text{train}}$ as the ID test set and the remainder as the train set. An illustration of the Eval-Fix strategy is provided in Figure 7.

Each class in the training set is required to be represented in both $\mathcal{D}^{\text{ID}}$ and $\mathcal{D}^{\text{OOD}}$, and vice versa. For all datasets, we generate three random splits and average metrics across these splits. We had to limit the number of splits to three due to the constrained number of available domains and the requirement for classes to be present in both the train and test splits. Each method is trained three times, and we report the average and 95%-confidence intervals calculated across model initializations.

Metrics. We evaluate Accuracy, F1-Score (Harmonic mean of precision and recall, useful in imbalanced datasets), ROC AUC (Area under the receiver operating characteristic curve), and ECE (Expected Calibration Error; reflects the reliability of the model's probability outputs)

Datasets. Our benchmark comprises 18 test datasets, 8 synthetic and 10 real-world. In addition, 12 validation datasets, 4 synthetic and the remaining 8 real-world, were used to optimize the hyperparameters of our approach via random search. While some of these datasets have been analyzed in previous work, there has been no comprehensive benchmark focusing on small tabular datasets undergoing distribution shifts. To address this gap, we carefully selected and generated a diverse range of datasets that exhibit temporal distribution shifts. The selected datasets originate from open dataset platforms or previous work in DG. Ground truth domain indices $c_k \in \mathcal{C}$ are known for synthetic datasets. For real-world datasets, we approximated domain indices $\hat{c_k}$ based on features that encode temporal information, which we transformed into discrete intervals. Also, some real-world datasets required subsampling due to their large size, which was beyond the current architecture of TabPFN. We provide full details, including descriptions for each dataset and pre-processing steps in Section A.10 of the Appendix.

Baseline Setup. Our baselines include state-of-the-art methods for tabular prediction. These include advanced Gradient Boosted Decision Trees like CatBoost [39], XGBoost [40], and LightGBM [41], which have demonstrated superior performance to standard neural network approaches in handling tabular data [11]. We also include TabPFN in its unmodified form (TabPFN$_{\text{base}}$; 13). Methods from the Wild-Time benchmark are examined separately and detailed in Section A.9.7 of the Appendix. All baseline methods besides TabPFN are subject to a time budget of 1,200 seconds on 8 CPUs and 1 GPU. For each method except TabPFN, which does not require tuning, a random hyperparameter search with 3-fold time series cross-validation was used. We chose the best-performing hyperparameters based on OOD ROC AUC within the allocated time.

Among our baselines, we considered three strategies:

1. Providing the full dataset $\mathcal{D}^{\text{train}}$ along with the corresponding domain indices $\mathcal{C}^{\text{train}}$ as a feature, aiming to allow for better reasoning of the shifts in the dataset (all dom. w. ind.).
2. Using the dataset without domain indices $\mathcal{D}^{\text{train}} = \{(\boldsymbol{x}_i^{\text{train}}, y_i^{\text{train}})\}_{i=1}^{n}$ (all dom. wo. ind.).
3. Limiting the training set to samples from the last training domain c_t. In this setting, we also omit the corresponding domain indices, resulting in the set $\mathcal{D}_{\hat{c}_t}^{\text{train}} = \{(\boldsymbol{x}_i^{\text{train}}, y_i^{\text{train}})\}_{i=1}^{n_{c_t}}$ (last dom. wo. ind.). The rationale behind the last scenario is to provide only training data closest to the subsequent test distribution. This strategy is not used for distribution shift baselines.

TabPFN Setup. For the TabPFN variants, both the original and our modified method (TabPFN$_{\text{dist}}$) were pre-trained for 30 epochs across 8 GPUs. This results in a total of 30,720,000 synthetically generated datasets processed during pre-training. While this pre-training step is moderately expensive, it is done offline, in advance, and only once as part of our algorithm development. Furthermore, the preprocessing parameters of both methods were optimized once on the validation datasets by random search over 300 configurations. We chose the configurations that yielded the best OOD ROC AUC performance. The resulting model and hyperparameters are used for all datasets, resulting in, on average, 110 times faster training and prediction time on our benchmark.

A.3 Ablations

Is our model's performance mostly based on Time2Vec preprocessing? To address this question, we conducted an ablation study where we trained a model with temporal domain indices normalized but not subjected to Time2Vec preprocessing (No T2V).

Table 1 presents the performance metrics of Drift-Resilient TabPFN, TabPFN-base, and our ablation model. The results indicate that while Time2Vec preprocessing may slightly improve model performance, it is statistically insignificant. Rather, the substantial performance improvements are largely due to our prior construction, used during the pre-training phase of the model. The decision to keep Time2Vec in the final model was guided by our HPO, which indicated a positive impact on average performance.

Table 1: Comparison of Drift-Resilient TabPFN with respect to the stated ablations. Metrics include ROC AUC and accuracy for both in-distribution (ID) and out-of-distribution (OOD) data.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
TabPFN$_{\text{dist}}$	all dom. w. ind.	$\mathbf{0.744}_{.018}$	$0.879_{.012}$	$\mathbf{0.689}_{.028}$	$0.837_{.022}$	$\mathbf{0.832}_{.018}$	$0.932_{.002}$	$\mathbf{0.091}_{.006}$	$0.074_{.014}$
No T2V	all dom. w. ind.	$0.742_{.004}$	$0.877_{.007}$	$0.685_{.002}$	$0.834_{.014}$	$\mathbf{0.832}_{.004}$	$0.931_{.009}$	$0.093_{.009}$	$0.071_{.005}$
TabPFN$_{\text{base}}$	all dom. w. ind.	$0.688_{.01}$	$\mathbf{0.885}_{.01}$	$0.62_{.012}$	$\mathbf{0.847}_{.017}$	$0.786_{.007}$	$\mathbf{0.935}_{.01}$	$0.119_{.006}$	$\mathbf{0.067}_{.005}$
	all dom. wo. ind.	$0.645_{.011}$	$0.852_{.016}$	$0.579_{.014}$	$0.801_{.02}$	$0.736_{.001}$	$0.914_{.007}$	$0.202_{.011}$	$0.076_{.007}$
	last dom. wo. ind.	$0.67_{.005}$	$0.867_{.004}$	$0.609_{.004}$	$0.823_{.011}$	$0.76_{.003}$	$0.915_{.019}$	$0.181_{.003}$	$0.128_{.007}$

A.4 Plots Illustrating Types of Distribution Shifts

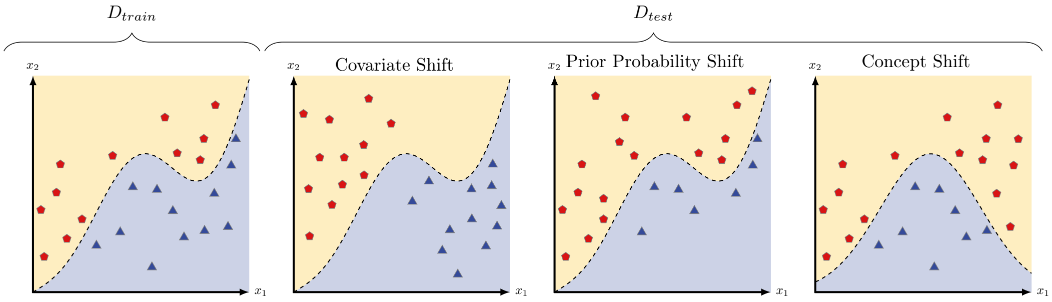


Figure 4: Illustration of initial and shifted data distributions alongside their optimal decision boundaries. The left panel depicts the initial classification dataset with two features and its true-data-optimal decision boundary. The right panel presents the dataset subjected to the three primary types of distribution shifts observed during test time.

A.5 Plots Illustrating Sampled Parameters of the Adjusted Prior

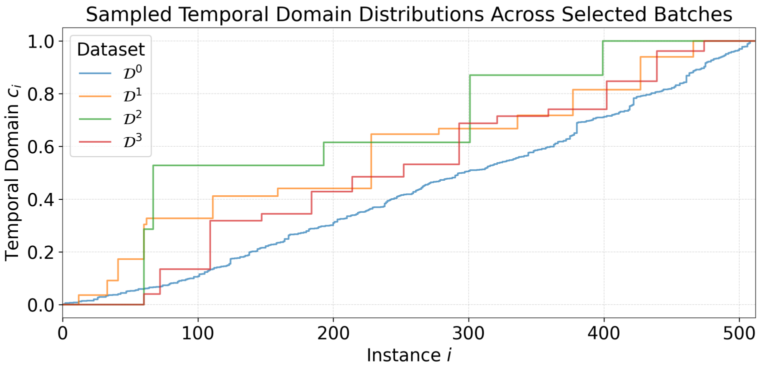


Figure 5: Share of temporal domains in exemplary datasets prior seen up to any instance i. The figure illustrates the range and structure of the sampled temporal domains $c_k \in \mathcal{C}$ across four representative datasets. It highlights variations in domain size and demonstrates the presence of arbitrary gaps, simulating irregularities in data sampling.

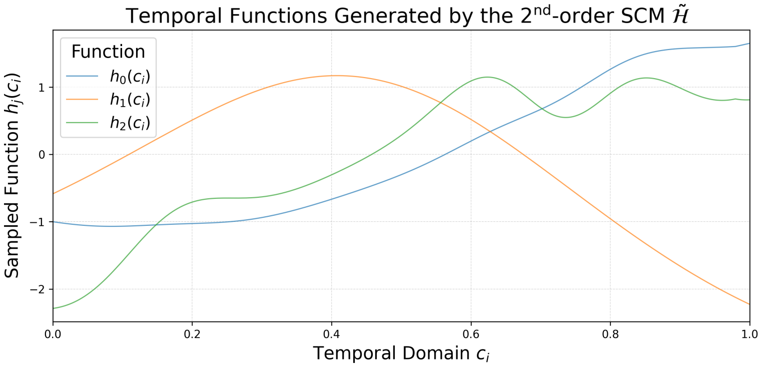


Figure 6: This figure presents three exemplary functions sampled from nodes within the network of a 2nd-order SCM $\tilde{\mathcal{H}}$. In the plot, the x-axis represents the input temporal domain $c_k \in \mathcal{C}$, while the y-axis displays the corresponding node activation.

A.6 Algorithmic Overview of Our Approach

Algorithm 1 This algorithm provides a high-level overview for generating a synthetic dataset in our prior. Although steps are depicted sequentially for clarity, many can be parallelized in actual implementation.

```
procedure SampleDataset
    $\mathcal{G} \leftarrow$ SampleSCM()                          ▷ Sample data-generating SCM
    $\tilde{\mathcal{G}} \leftarrow \mathcal{G}$.Expand()                          ▷ Expand to functional representation

    $\mathcal{H} \leftarrow$ SampleSCM()                          ▷ Sample 2nd-order SCM
    $\tilde{\mathcal{H}} \leftarrow \mathcal{H}$.Expand()                          ▷ Expand to functional representation

    $\mathcal{C} \leftarrow \{c_1, c_2, \ldots, c_t\}$                    ▷ Sample temporal domains
    $\mathcal{D} \leftarrow \emptyset$                                    ▷ Initialize dataset

    for all $c_k \in \mathcal{C}$ do
        $\boldsymbol{\omega}_{c_k} \leftarrow \tilde{\mathcal{H}}$.Forward($c_k$)                  ▷ Sample edge shifts
        $\tilde{\mathcal{G}}_{c_k} \leftarrow \tilde{\mathcal{G}}$.Update($\boldsymbol{\omega}_{c_k}$)                  ▷ Update edge weights

        $\mathcal{D}_{c_k} \leftarrow \emptyset$                              ▷ Initialize sub-dataset
        for all $i \in \{1, ..., n_{c_k}\}$ do                  ▷ Sample sub-dataset
            $(\boldsymbol{x}_i, y_i, c_k) \leftarrow \tilde{\mathcal{G}}_{c_k}$.Forward($\epsilon_i$)
            $\mathcal{D}_{c_k} \leftarrow \mathcal{D}_{c_k} \cup \{(\boldsymbol{x}_i, y_i, c_k)\}$
        end for

        $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{c_k}$                              ▷ Extend dataset
    end for

    return $\mathcal{D}$                                  ▷ Return dataset
end procedure
```

A.7 Detailed Overview of Wild-Time Methods

A.7.1 Classical Supervised Learning

Empirical Risk Minimization (ERM). ERM - a fundamental approach in supervised learning - focuses on minimizing the average loss over the training dataset. In Wild-Time ERM is defined as typical supervised learning without making use of any temporal information.

A.7.2 Continual Learning

Fine-Tuning (FT). FT extends the ERM approach by training on the data of each successive temporal domain separately, allowing the model to adapt to new distributions but risking catastrophic forgetting of past tasks.

Elastic Weight Consolidation (EWC). EWC [42] counters catastrophic forgetting by adding a regularization term that constrains the changes to important model parameters, thus preserving knowledge from previous tasks.

Synaptic Intelligence (SI). SI [43] captures a synaptic strength metric over time for each model parameter. This metric is used as a regularizer to limit changes to important parameters during learning.

Averaged Gradient Episodic Memory (A-GEM). A-GEM [44] maintains a small episodic memory and computes gradients not just for the current task but also the average of the gradients over several past tasks stored in the episodic memory.

A.7.3 Temporally Invariant Learning

Deep Correlation Alignment (Deep CORAL). Initially developed for domain adaptation, Deep CORAL [23] aims to align the second-order statistics of features between the source and target domains to minimize distribution shift. In the Wild-Time benchmark, this original purpose is modified to align features across different temporal domains within the training set, thus converting it into a DG method. For handling temporal shifts, it is extended into CORAL-T, which employs sliding windows to segment the data stream into temporal substreams, treating each as a separate domain for alignment. [5]

Group Distributionally Robust Optimization (GroupDRO). Originally designed at optimizing on the worst-performing group within the training data, GroupDRO [20] aims to learn a model that is robust across varying group distributions. In the Wild-Time benchmark, this method is adapted to the temporal context as GroupDRO-T. It utilizes sliding window-based segmentation to create temporal substreams, treating each as a separate group for distributionally robust optimization. [5]

Invariant Risk Minimization (IRM). IRM [19] aims to identify a data representation that is consistently predictive across different domains. In the context of Wild-Time, the method is adapted to temporal shifts and named IRM-T. It employs sliding window-based segmentation to create temporal substreams, which are then treated as individual domains for invariant risk minimization. [5]

Mixup. Mixup [21] is an interpolation-based data augmentation technique that creates new training examples by blending the features and labels of existing samples. This technique aims to enhance the model's ability to generalize across domains by diversifying the training data. It replaces the original training samples with these newly generated interpolations for more robust training.

Learning with Selective Augmentation (LISA). LISA [22], motivated by Mixup, employs selective interpolation to neutralize domain-specific information in the training data. It comes in two variants: intra-label LISA, which interpolates examples from different domains but having the same label, and intra-domain LISA, which interpolates examples within the same domain but having different labels. In Wild-Time, only intra-label LISA is used [5].

A.7.4 Self-Supervised Learning

Simple Framework for Contrastive Learning of Visual Representations (SimCLR). SimCLR [45] employs contrastive learning to maximize the agreement between different augmentations of the same image, thereby enhancing the quality of learned visual representations. The approach benefits from learnable nonlinear transformations and optimized contrastive loss parameters.

Swapping Assignments between multiple Views of the same image (SwAV). SwAV [46] employs a clustering approach within the contrastive learning framework. It enforces consistency between cluster assignments across different augmentations of the same image. This obviates the need for pairwise feature comparisons, offering computational efficiency.

A.7.5 Bayesian Learning

Stochastic Weight Averaging (SWA). SWA [34] averages multiple parameter values along the stochastic gradient descent (SGD) trajectory to improve in-distribution generalization. It operates with minimal computational overhead and aims to approximate the posterior distribution over model parameters, reflected in the flatness of the learned optima. [5]

A.8 Illustration of Adopted Evaluation Strategy

In Figure 7, we provide an illustration of the Eval-Fix evaluation strategy, originally proposed by Yao et al. [5] in the context of the Wild-Time benchmark. It should be noted that the formalism has been adapted to align with the notation used in this paper.

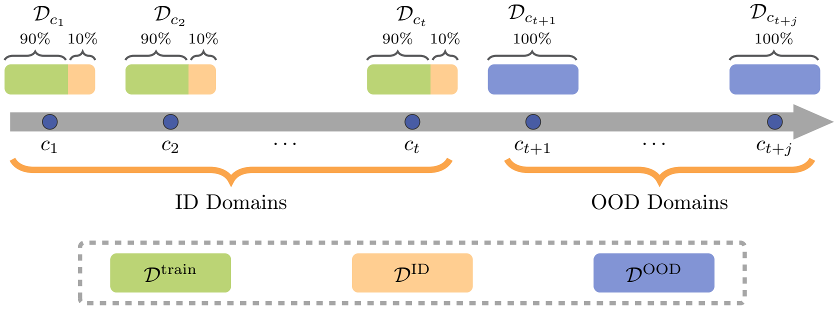


Figure 7: Adapted from the Wild-Time benchmark [5], this illustration portrays the Eval-Fix evaluation strategy employed in our study. The domain boundary is indicated by c_t, beyond which datasets are considered part of the out-of-distribution (OOD) test set $\mathcal{D}^{\text{OOD}}$. To evaluate in-distribution (ID) performance, we subsample 10% of the samples from each dataset prior to this boundary, forming the datasets $\mathcal{D}^{\text{train}}$ and $\mathcal{D}^{\text{ID}}$.

A.9 Additional Experiments

A.9.1 Perturbation of Temporal Domain Indices within the Prior

This additional experiment analyzes the impact of perturbing temporal domain indices on the performance of our Drift-Resilient TabPFN model. In real-world datasets, ground truth temporal domain information is often unknown and must be approximated, creating a crucial difference between these datasets and those generated by the prior in our methodology. To assess this, we conducted an experiment during model development wherein we intentionally modified the ground truth domain indices $\mathcal{C}$ during the pre-training phase prior to feeding them into the transformer.

In this context, we explored three primary techniques:

Shifting Domain Boundaries Probabilistically. We shifted the boundaries of domains and reported instances near those boundaries as belonging to adjacent domains.

Merging Domains. Multiple domains were probabilistically merged into a single domain, thereby introducing ambiguity in domain information.

Noise Injection Random noise was added to each domain indicator, further complicating reasoning about temporal distribution shifts based on these indicators.

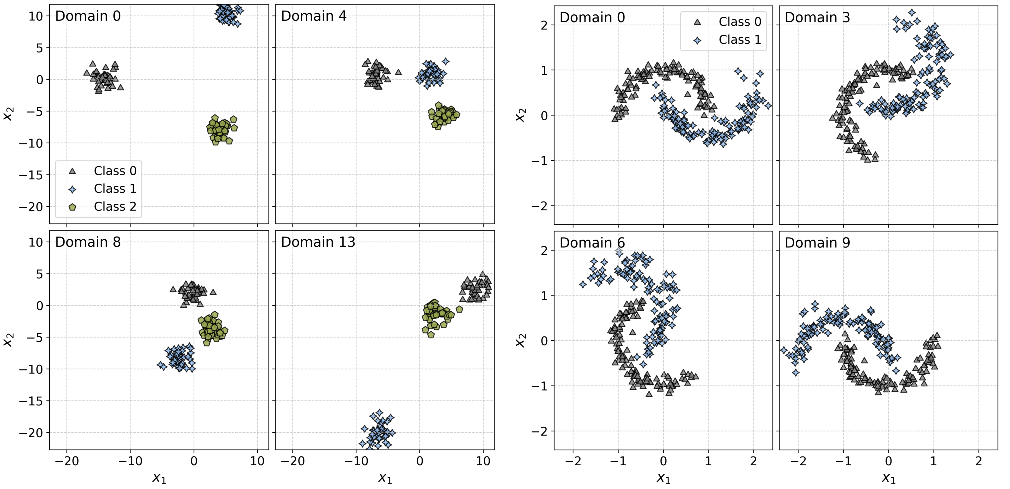
Our experiments on our validation datasets, listed in Table 2, show that overall these perturbations adversely affect model performance. The findings suggest that the model requires ground-truth domain indices for effective training, emphasizing the importance of accurate domain information in real-world applications.

Table 2: Performance evaluation of Drift-Resilient TabPFN against models with perturbed domain information. Three techniques were examined: (1) Shifting domain boundaries probabilistically, reporting instances near those boundaries as belonging to the other domain; (2) Probabilistically merging multiple domains into one; (3) Adding noise to each domain indicator. Our results indicate that perturbed domain indices overall led to a decline in model performance, emphasizing the model's requirement for ground-truth domain indices during training.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
TabPFN$_{\textbf{dist}}$	no changes	0.74	**0.867**	**0.692**	**0.832**	**0.837**	**0.908**	**0.085**	**0.076**
TabPFN$_{\textbf{pert. dom.}}$	alt dom. bound.	0.734	0.86	0.682	0.822	0.829	0.905	0.092	0.078
	alt dom. bound. / merge dom.	**0.742**	0.857	0.69	0.815	0.833	0.906	0.09	0.079
	alt dom. bound. / merge dom. / noise	0.704	0.835	0.64	0.78	0.792	0.888	0.101	0.108

A.9.2 Qualitative Analysis: Overview of the Shifts in the Datasets Analyzed

This section offers plots of the Intersecting Blobs and Rotated Two Moons datasets across specific temporal domains. The Intersecting Blobs dataset is visualized in Figure 8a for domains $\mathcal{C} = \{0, 4, 8, 13\}$. The Rotated Two Moons dataset is presented in Figure 8b for domains $\mathcal{C} = \{0, 3, 6, 9\}$.



(a) Intersecting Blobs

(b) Rotated Two Moons

Figure 8: This figure shows the temporal shifts of two synthetic datasets across selected domains.

A.9.3 Qualitative Analysis: Decision Boundaries on Rotated Two Moons Dataset

In addition to the qualitative analysis we conducted for the Intersecting Blobs dataset in the main text of this work, this subsection provides additional illustrations of the Rotated Two Moons dataset. The corresponding visualizations for our approach as well as the TabPFN baseline are provided in Figure 9.

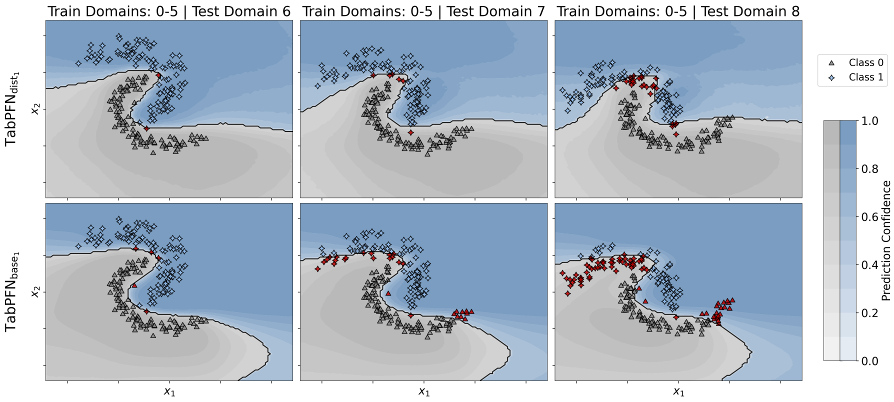


Figure 9: This figure contrasts the predictive behavior of $\text{TabPFN}_{\text{dist}}$ and $\text{TabPFN}_{\text{base}}$ on the Rotated Two Moons dataset. It illustrates how each model adapts to different testing domains when trained on domains $\mathcal{C}^{\text{train}} = \{0, 1, 2, 3, 4, 5\}$. The color shading indicates the maximum class probability at each point, with decision boundaries shown when this probability exceeds 50%. Incorrectly classified samples are highlighted in red.

A.9.4 Quantitative Analysis: Performance on Combined, Synthetic, and Real-World Datasets

Table 3: Comparison of Drift-Resilient TabPFN with various baselines and settings across the combined **real-world** and **synthetic** datasets. Metrics include accuracy, F1, ROC, and ECE for both in-distribution (ID) and out-of-distribution (OOD) data, averaged over three initializations and reported with 95% confidence intervals. The best mean of each metric is marked in bold. Metric arrows indicate optimization direction.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
$\text{TabPFN}_{\text{dist}}$	all dom. w. ind.	$\mathbf{0.744}_{.018}$	$0.879_{.012}$	$\mathbf{0.689}_{.028}$	$0.837_{.022}$	$\mathbf{0.832}_{.018}$	$0.932_{.002}$	$\mathbf{0.091}_{.006}$	$0.074_{.014}$
$\text{TabPFN}_{\text{base}}$	all dom. w. ind.	$0.688_{.01}$	$\mathbf{0.885}_{.01}$	$0.62_{.012}$	$\mathbf{0.847}_{.017}$	$0.786_{.007}$	$\mathbf{0.935}_{.01}$	$0.119_{.006}$	$\mathbf{0.067}_{.005}$
	all dom. wo. ind.	$0.645_{.011}$	$0.852_{.016}$	$0.579_{.014}$	$0.801_{.02}$	$0.736_{.001}$	$0.914_{.007}$	$0.202_{.011}$	$0.076_{.007}$
	last dom. wo. ind.	$0.67_{.005}$	$0.867_{.004}$	$0.609_{.004}$	$0.823_{.011}$	$0.76_{.003}$	$0.915_{.019}$	$0.181_{.003}$	$0.128_{.007}$
CatBoost	all dom. w. ind.	$0.677_{.006}$	$0.874_{.007}$	$0.62_{.005}$	$0.836_{.01}$	$0.766_{.003}$	$0.919_{.011}$	$0.222_{.007}$	$0.084_{.009}$
	all dom. wo. ind.	$0.632_{.003}$	$0.836_{.013}$	$0.568_{.005}$	$0.781_{.009}$	$0.714_{.012}$	$0.894_{.014}$	$0.24_{.02}$	$0.097_{.015}$
	last dom. wo. ind.	$0.657_{.002}$	$0.852_{.014}$	$0.599_{.004}$	$0.811_{.024}$	$0.722_{.005}$	$0.907_{.01}$	$0.256_{.006}$	$0.133_{.012}$
XGBoost	all dom. w. ind.	$0.664_{.005}$	$0.859_{.004}$	$0.61_{.013}$	$0.828_{.003}$	$0.754_{.006}$	$0.9_{.019}$	$0.194_{.018}$	$0.111_{.02}$
	all dom. wo. ind.	$0.633_{.035}$	$0.831_{.024}$	$0.568_{.033}$	$0.778_{.031}$	$0.718_{.033}$	$0.881_{.028}$	$0.194_{.054}$	$0.12_{.042}$
	last dom. wo. ind.	$0.664_{.01}$	$0.824_{.023}$	$0.599_{.016}$	$0.758_{.054}$	$0.733_{.009}$	$0.887_{.016}$	$0.199_{.024}$	$0.167_{.011}$
LightGBM	all dom. w. ind.	$0.65_{.009}$	$0.842_{.024}$	$0.594_{.008}$	$0.8_{.024}$	$0.738_{.008}$	$0.908_{.008}$	$0.198_{.009}$	$0.093_{.016}$
	all dom. wo. ind.	$0.625_{.02}$	$0.832_{.019}$	$0.561_{.018}$	$0.773_{.018}$	$0.706_{.009}$	$0.888_{.01}$	$0.199_{.009}$	$0.097_{.005}$
	last dom. wo. ind.	$0.629_{.02}$	$0.797_{.009}$	$0.542_{.031}$	$0.72_{.028}$	$0.686_{.006}$	$0.852_{.018}$	$0.224_{.011}$	$0.149_{.016}$
Wild-Time ERM	all dom. w. ind.	$0.627_{.036}$	$0.821_{.032}$	$0.525_{.052}$	$0.771_{.044}$	$0.688_{.028}$	$0.877_{.025}$	$0.263_{.054}$	$0.108_{.017}$
	all dom. wo. ind.	$0.582_{.028}$	$0.803_{.01}$	$0.519_{.028}$	$0.746_{.018}$	$0.673_{.02}$	$0.862_{.008}$	$0.255_{.019}$	$0.104_{.015}$
	last dom. wo. ind.	$0.587_{.026}$	$0.784_{.03}$	$0.528_{.034}$	$0.74_{.011}$	$0.666_{.018}$	$0.843_{.02}$	$0.323_{.038}$	$0.19_{.025}$
Wild-Time SWA	all dom. w. ind.	$0.627_{.047}$	$0.824_{.003}$	$0.534_{.069}$	$0.777_{.014}$	$0.685_{.036}$	$0.873_{.015}$	$0.274_{.053}$	$0.11_{.007}$
	all dom. wo. ind.	$0.588_{.025}$	$0.802_{.007}$	$0.53_{.029}$	$0.748_{.008}$	$0.681_{.03}$	$0.863_{.008}$	$0.262_{.036}$	$0.109_{.01}$

Table 4: Comparison of Drift-Resilient TabPFN with various baselines and settings across the subset of **synthetic** datasets. Metrics include accuracy, F1, ROC, and ECE for both in-distribution (ID) and out-of-distribution (OOD) data, averaged over three initializations and reported with 95% confidence intervals. The best mean of each metric is marked in bold. Metric arrows indicate optimization direction.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
TabPFN$_{\text{dist}}$	all dom. w. ind.	**0.754** $_{.032}$	0.959 $_{.011}$	**0.697** $_{.048}$	0.935 $_{.033}$	**0.844** $_{.03}$	0.987 $_{.002}$	**0.126** $_{.018}$	0.038 $_{.003}$
TabPFN$_{\text{base}}$	all dom. w. ind.	0.658 $_{.018}$	**0.963** $_{.006}$	0.567 $_{.015}$	0.935 $_{.02}$	0.749 $_{.017}$	0.986 $_{.007}$	0.164 $_{.014}$	**0.029** $_{.003}$
	all dom. wo. ind.	0.571 $_{.014}$	0.901 $_{.006}$	0.467 $_{.016}$	0.848 $_{.009}$	0.631 $_{.01}$	0.955 $_{.003}$	0.322 $_{.016}$	0.053 $_{.005}$
	last dom. wo. ind.	0.651 $_{.002}$	0.939 $_{.015}$	0.574 $_{.002}$	0.918 $_{.031}$	0.727 $_{.006}$	0.975 $_{.001}$	0.27 $_{.011}$	0.066 $_{.001}$
CatBoost	all dom. w. ind.	0.665 $_{.008}$	0.958 $_{.003}$	0.588 $_{.008}$	**0.94** $_{.009}$	0.73 $_{.01}$	**0.989** $_{.002}$	0.297 $_{.006}$	0.037 $_{.006}$
	all dom. wo. ind.	0.575 $_{.006}$	0.885 $_{.003}$	0.476 $_{.008}$	0.831 $_{.002}$	0.613 $_{.013}$	0.942 $_{.007}$	0.325 $_{.006}$	0.063 $_{.014}$
	last dom. wo. ind.	0.639 $_{.004}$	0.932 $_{.017}$	0.564 $_{.005}$	0.916 $_{.021}$	0.684 $_{.005}$	0.962 $_{.012}$	0.301 $_{.019}$	0.065 $_{.006}$
XGBoost	all dom. w. ind.	0.645 $_{.018}$	0.936 $_{.012}$	0.57 $_{.019}$	0.931 $_{.005}$	0.705 $_{.011}$	0.968 $_{.02}$	0.253 $_{.032}$	0.075 $_{.013}$
	all dom. wo. ind.	0.582 $_{.07}$	0.872 $_{.031}$	0.48 $_{.074}$	0.818 $_{.039}$	0.621 $_{.06}$	0.926 $_{.035}$	0.245 $_{.088}$	0.097 $_{.034}$
	last dom. wo. ind.	0.645 $_{.008}$	0.916 $_{.009}$	0.565 $_{.009}$	0.88 $_{.019}$	0.688 $_{.017}$	0.956 $_{.012}$	0.256 $_{.018}$	0.111 $_{.003}$
LightGBM	all dom. w. ind.	0.646 $_{.016}$	0.943 $_{.007}$	0.57 $_{.015}$	0.927 $_{.015}$	0.687 $_{.013}$	0.982 $_{.002}$	0.281 $_{.008}$	0.056 $_{.011}$
	all dom. wo. ind.	0.581 $_{.01}$	0.884 $_{.005}$	0.482 $_{.007}$	0.829 $_{.005}$	0.617 $_{.016}$	0.935 $_{.001}$	0.273 $_{.01}$	0.069 $_{.017}$
	last dom. wo. ind.	0.629 $_{.004}$	0.917 $_{.003}$	0.553 $_{.006}$	0.892 $_{.018}$	0.662 $_{.005}$	0.958 $_{.007}$	0.288 $_{.012}$	0.077 $_{.007}$
Wild-Time ERM	all dom. w. ind.	0.648 $_{.046}$	0.945 $_{.017}$	0.489 $_{.092}$	0.906 $_{.026}$	0.65 $_{.042}$	0.973 $_{.021}$	0.304 $_{.038}$	0.041 $_{.006}$
	all dom. wo. ind.	0.576 $_{.021}$	0.885 $_{.04}$	0.487 $_{.035}$	0.837 $_{.049}$	0.621 $_{.03}$	0.943 $_{.015}$	0.282 $_{.012}$	0.058 $_{.024}$
	last dom. wo. ind.	0.632 $_{.006}$	0.921 $_{.017}$	0.566 $_{.007}$	0.9 $_{.035}$	0.688 $_{.01}$	0.962 $_{.005}$	0.282 $_{.018}$	0.094 $_{.016}$
Wild-Time SWA	all dom. w. ind.	0.636 $_{.05}$	0.923 $_{.034}$	0.489 $_{.088}$	0.877 $_{.06}$	0.651 $_{.035}$	0.958 $_{.03}$	0.313 $_{.046}$	0.054 $_{.015}$
	all dom. wo. ind.	0.573 $_{.032}$	0.887 $_{.019}$	0.48 $_{.042}$	0.837 $_{.017}$	0.631 $_{.043}$	0.943 $_{.012}$	0.3 $_{.055}$	0.059 $_{.002}$

Table 5: Comparison of Drift-Resilient TabPFN with various baselines and settings across the subset of **real-world** datasets. Metrics include accuracy, F1, ROC, and ECE for both in-distribution (ID) and out-of-distribution (OOD) data, averaged over three initializations and reported with 95% confidence intervals. The best mean of each metric is marked in bold. Metric arrows indicate optimization direction.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
TabPFN$_{\text{dist}}$	all dom. w. ind.	**0.736** $_{.007}$	0.814 $_{.014}$	**0.682** $_{.012}$	0.759 $_{.015}$	**0.822** $_{.01}$	0.887 $_{.006}$	**0.062** $_{.004}$	0.103 $_{.026}$
TabPFN$_{\text{base}}$	all dom. w. ind.	0.712 $_{.012}$	**0.822** $_{.013}$	0.661 $_{.022}$	**0.777** $_{.018}$	0.816 $_{.006}$	**0.894** $_{.013}$	0.083 $_{.001}$	0.097 $_{.007}$
	all dom. wo. ind.	0.704 $_{.01}$	0.813 $_{.023}$	0.668 $_{.014}$	0.764 $_{.03}$	0.82 $_{.006}$	0.882 $_{.011}$	0.106 $_{.019}$	**0.095** $_{.011}$
	last dom. wo. ind.	0.685 $_{.008}$	0.809 $_{.016}$	0.637 $_{.005}$	0.746 $_{.036}$	0.787 $_{.008}$	0.867 $_{.036}$	0.109 $_{.005}$	0.177 $_{.012}$
CatBoost	all dom. w. ind.	0.687 $_{.005}$	0.807 $_{.012}$	0.646 $_{.003}$	0.753 $_{.017}$	0.796 $_{.004}$	0.862 $_{.019}$	0.161 $_{.016}$	0.122 $_{.019}$
	all dom. wo. ind.	0.677 $_{.008}$	0.797 $_{.025}$	0.642 $_{.005}$	0.741 $_{.015}$	0.794 $_{.011}$	0.856 $_{.022}$	0.172 $_{.032}$	0.125 $_{.037}$
	last dom. wo. ind.	0.671 $_{.002}$	0.788 $_{.023}$	0.627 $_{.004}$	0.728 $_{.05}$	0.752 $_{.007}$	0.863 $_{.022}$	0.221 $_{.017}$	0.188 $_{.023}$
XGBoost	all dom. w. ind.	0.68 $_{.008}$	0.797 $_{.011}$	0.642 $_{.01}$	0.745 $_{.004}$	0.793 $_{.01}$	0.845 $_{.02}$	0.147 $_{.027}$	0.141 $_{.027}$
	all dom. wo. ind.	0.674 $_{.025}$	0.798 $_{.019}$	0.639 $_{.004}$	0.746 $_{.026}$	0.795 $_{.011}$	0.845 $_{.024}$	0.153 $_{.028}$	0.138 $_{.049}$
	last dom. wo. ind.	0.679 $_{.014}$	0.75 $_{.041}$	0.626 $_{.034}$	0.66 $_{.109}$	0.769 $_{.008}$	0.833 $_{.022}$	0.154 $_{.028}$	0.212 $_{.021}$
LightGBM	all dom. w. ind.	0.654 $_{.015}$	0.761 $_{.042}$	0.614 $_{.01}$	0.698 $_{.031}$	0.778 $_{.008}$	0.848 $_{.013}$	0.133 $_{.017}$	0.123 $_{.02}$
	all dom. wo. ind.	0.66 $_{.029}$	0.79 $_{.031}$	0.624 $_{.028}$	0.728 $_{.029}$	0.778 $_{.003}$	0.849 $_{.018}$	0.14 $_{.009}$	0.12 $_{.018}$
	last dom. wo. ind.	0.629 $_{.036}$	0.701 $_{.019}$	0.533 $_{.055}$	0.582 $_{.046}$	0.706 $_{.008}$	0.768 $_{.03}$	0.172 $_{.014}$	0.206 $_{.025}$
Wild-Time ERM	all dom. w. ind.	0.61 $_{.037}$	0.721 $_{.056}$	0.553 $_{.046}$	0.664 $_{.059}$	0.719 $_{.017}$	0.8 $_{.042}$	0.23 $_{.07}$	0.161 $_{.035}$
	all dom. wo. ind.	0.587 $_{.033}$	0.737 $_{.014}$	0.545 $_{.028}$	0.673 $_{.013}$	0.714 $_{.012}$	0.798 $_{.024}$	0.233 $_{.043}$	0.14 $_{.008}$
	last dom. wo. ind.	0.551 $_{.041}$	0.674 $_{.045}$	0.498 $_{.067}$	0.612 $_{.03}$	0.648 $_{.039}$	0.749 $_{.037}$	0.355 $_{.061}$	0.267 $_{.034}$
Wild-Time SWA	all dom. w. ind.	0.62 $_{.049}$	0.745 $_{.028}$	0.57 $_{.055}$	0.697 $_{.046}$	0.712 $_{.039}$	0.805 $_{.007}$	0.242 $_{.059}$	0.155 $_{.021}$
	all dom. wo. ind.	0.6 $_{.019}$	0.733 $_{.017}$	0.57 $_{.019}$	0.677 $_{.004}$	0.721 $_{.029}$	0.798 $_{.008}$	0.232 $_{.021}$	0.15 $_{.017}$

A.9.5 Quantitative Analysis: Critical Difference Diagrams

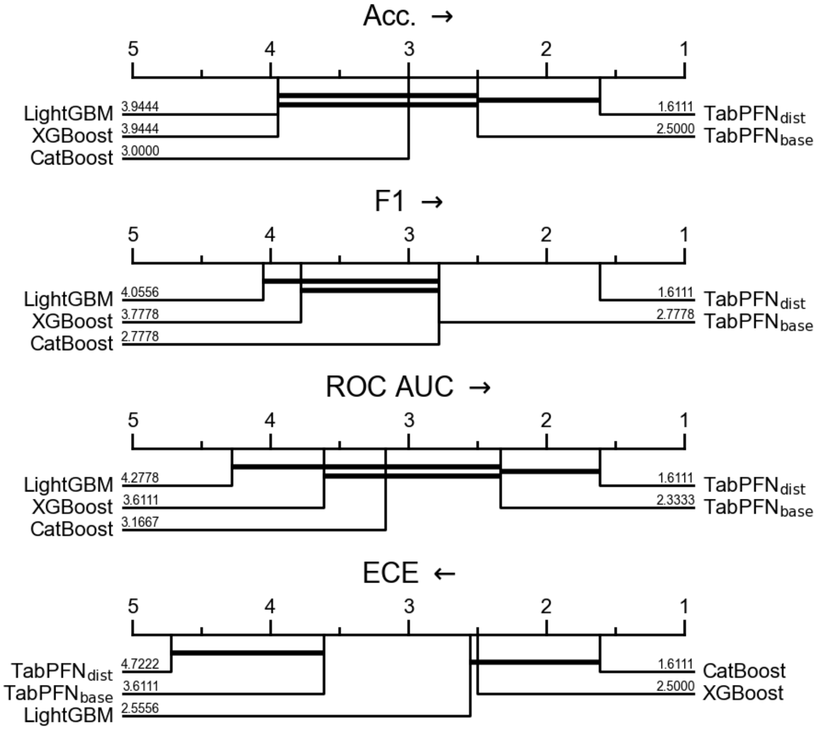


Figure 10: This figure presents critical difference diagrams of our evaluated metrics on OOD data, analyzed using the Wilcoxon-Holm method [47–50] across the best performing OOD models evaluated in this work. The diagrams indicate significant differences of Drift-Resilient TabPFN against the tree-based methods. For the F1 metric our method shows significant differences against all top performing baselines. Arrows indicate optimization direction.

A.9.6 Analyzing the Relationship Between Out-of-Distribution Performance and Difficulty Across Datasets

In this supplementary evaluation, we investigate the correlation between the inherent difficulty of a dataset concerning distribution shifts and the performance of our approach in comparison to the baselines.

Defining Difficulty Metrics. Our first step involves assigning a difficulty score to individual dataset splits, using a basic XGBoost model that doesn't include domain indices as a feature of the dataset. This score is determined using the formula $ID - OOD$, where both ID and OOD performances are measured using ROC AUC. This difficulty score quantifies the decline in model performance when transitioning from ID to OOD data. After calculating the difficulty scores for each dataset split, we proceed to train each of the evaluated methods across all domains, including the domain indices. We then compute the performance gap, which is also calculated as $ID - OOD$, for each method.

These metrics are then visualized by plotting the dataset's difficulty score against its corresponding performance gap for each dataset split. We also fit linear regression lines to the scatter plots representing each method. The final plot is illustrated in Figure 11.

Although the data points cannot be interpreted directly, the linear regression suggests that our method experiences the least decline in ROC AUC performance as dataset difficulty increases, affirming its robustness. The TabPFN baseline is the second-best performer, while the other models show more significant drops in performance.

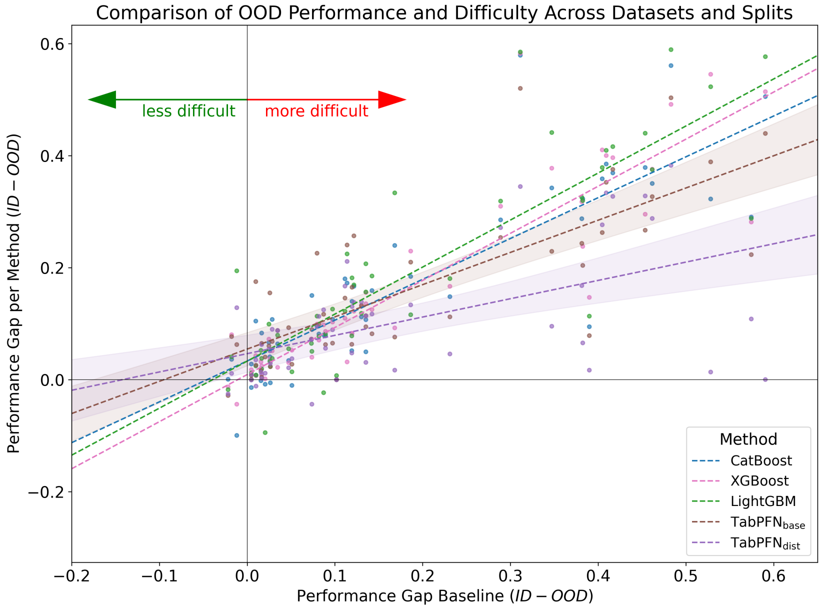


Figure 11: This figure illustrates a comparative analysis of the resilience of the listed methods to out-of-distribution (OOD) difficulty across multiple datasets and splits. The x-axis captures the difficulty of each dataset split, while the y-axis measures the performance drop of a method compared to in-distribution (ID) performance. Individual methods are represented by scatter points and their corresponding linear regression lines, with shaded regions indicating the 95% confidence intervals for TabPFN methods. Directional arrows signify increasing or decreasing dataset difficulty. Flatter regression slopes indicate models that are more resilient to increases in dataset difficulty due to distribution shifts.

A.9.7 Quantitative Analysis: Comparison Against Wild-Time Methods

This section offers a quantitative evaluation of our model against methods found in the Wild-Time benchmark [5]. We focus exclusively on methods applicable to tabular data, omitting SimCLR and SwAV which are specifically designed for image datasets. Detailed explanations of these methods are available in Section A.7.

As a base model, we used a Multilayer Perceptron (MLP) optimized through Hyperparameter Optimization (HPO).

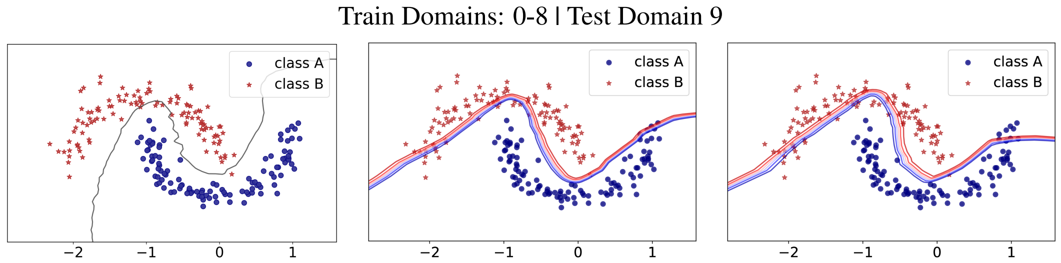
The findings of this quantitative comparison are presented in Table 6. Notably, none of the evaluated Wild-Time methods demonstrated performance equal with our approach or the other baseline methods on OOD data. This discrepancy is likely due to the small number of instances in the datasets used in our evaluations, which affects the generalization of these deep learning techniques. Among the Wild-Time methods, SWA emerged as the most effective in handling OOD data.

Table 6: Comparison of Drift-Resilient TabPFN with the applicable baselines of the Wild-Time benchmark [5] across the combined **real-world** and **synthetic** datasets. Metrics include accuracy, F1, ROC, and ECE for both in-distribution (ID) and out-of-distribution (OOD) data, averaged over three initializations and reported with 95% confidence intervals. The best mean of each metric is marked in bold. Metric arrows indicate optimization direction.

Model	**Variant**	**Acc. ↑**		**F1 ↑**		**ROC ↑**		**ECE ↓**	
		OOD	ID	OOD	ID	OOD	ID	OOD	ID
TabPFN$_{\text{dist}}$	all dom. w. ind.	**0.744** $_{.018}$	**0.879** $_{.012}$	**0.689** $_{.028}$	**0.837** $_{.022}$	**0.832** $_{.018}$	**0.932** $_{.002}$	**0.091** $_{.006}$	**0.074** $_{.014}$
ERM	all dom. w. ind.	0.627 $_{.036}$	0.821 $_{.032}$	0.525 $_{.052}$	0.771 $_{.044}$	0.688 $_{.028}$	0.877 $_{.025}$	0.263 $_{.054}$	0.108 $_{.017}$
	all dom. wo. ind.	0.582 $_{.028}$	0.803 $_{.01}$	0.519 $_{.028}$	0.746 $_{.018}$	0.673 $_{.02}$	0.862 $_{.008}$	0.255 $_{.019}$	0.104 $_{.015}$
	last dom. wo. ind.	0.587 $_{.026}$	0.784 $_{.03}$	0.528 $_{.034}$	0.74 $_{.011}$	0.666 $_{.018}$	0.843 $_{.02}$	0.323 $_{.038}$	0.19 $_{.025}$
FT	all dom. w. ind.	0.51 $_{.031}$	0.645 $_{.014}$	0.422 $_{.046}$	0.559 $_{.033}$	0.594 $_{.027}$	0.693 $_{.039}$	0.357 $_{.013}$	0.256 $_{.007}$
	all dom. wo. ind.	0.544 $_{.038}$	0.677 $_{.034}$	0.481 $_{.046}$	0.604 $_{.03}$	0.65 $_{.025}$	0.756 $_{.031}$	0.33 $_{.03}$	0.231 $_{.061}$
EWC	all dom. w. ind.	0.498 $_{.026}$	0.621 $_{.034}$	0.405 $_{.038}$	0.522 $_{.042}$	0.569 $_{.027}$	0.668 $_{.006}$	0.342 $_{.006}$	0.239 $_{.034}$
	all dom. wo. ind.	0.518 $_{.023}$	0.686 $_{.045}$	0.456 $_{.032}$	0.617 $_{.051}$	0.603 $_{.012}$	0.74 $_{.026}$	0.309 $_{.048}$	0.195 $_{.005}$
SI	all dom. w. ind.	0.478 $_{.057}$	0.63 $_{.006}$	0.383 $_{.089}$	0.526 $_{.015}$	0.566 $_{.061}$	0.673 $_{.023}$	0.37 $_{.046}$	0.238 $_{.012}$
	all dom. wo. ind.	0.495 $_{.019}$	0.674 $_{.035}$	0.425 $_{.036}$	0.588 $_{.035}$	0.597 $_{.022}$	0.744 $_{.03}$	0.346 $_{.002}$	0.214 $_{.049}$
A-GEM	all dom. w. ind.	0.513 $_{.031}$	0.677 $_{.038}$	0.405 $_{.04}$	0.576 $_{.045}$	0.594 $_{.061}$	0.741 $_{.038}$	0.367 $_{.024}$	0.223 $_{.021}$
	all dom. wo. ind.	0.486 $_{.035}$	0.684 $_{.023}$	0.403 $_{.061}$	0.583 $_{.05}$	0.58 $_{.035}$	0.747 $_{.019}$	0.364 $_{.073}$	0.221 $_{.054}$
CORAL-T	all dom. w. ind.	0.579 $_{.051}$	0.777 $_{.02}$	0.481 $_{.085}$	0.714 $_{.041}$	0.637 $_{.036}$	0.837 $_{.016}$	0.252 $_{.058}$	0.143 $_{.025}$
	all dom. wo. ind.	0.569 $_{.032}$	0.771 $_{.035}$	0.495 $_{.029}$	0.702 $_{.039}$	0.643 $_{.016}$	0.837 $_{.021}$	0.232 $_{.027}$	0.15 $_{.003}$
GroupDRO-T	all dom. w. ind.	0.58 $_{.025}$	0.779 $_{.014}$	0.503 $_{.038}$	0.723 $_{.009}$	0.642 $_{.036}$	0.84 $_{.007}$	0.285 $_{.019}$	0.125 $_{.01}$
	all dom. wo. ind.	0.576 $_{.025}$	0.775 $_{.023}$	0.514 $_{.023}$	0.725 $_{.036}$	0.658 $_{.008}$	0.843 $_{.006}$	0.264 $_{.076}$	0.135 $_{.04}$
IRM-T	all dom. w. ind.	0.577 $_{.021}$	0.746 $_{.014}$	0.49 $_{.051}$	0.689 $_{.013}$	0.633 $_{.021}$	0.819 $_{.016}$	0.259 $_{.03}$	0.12 $_{.013}$
	all dom. wo. ind.	0.559 $_{.018}$	0.742 $_{.023}$	0.498 $_{.031}$	0.678 $_{.047}$	0.644 $_{.005}$	0.822 $_{.013}$	0.252 $_{.031}$	0.134 $_{.011}$
Mixup	all dom. w. ind.	0.617 $_{.028}$	0.8 $_{.023}$	0.521 $_{.016}$	0.702 $_{.008}$	0.681 $_{.029}$	0.859 $_{.021}$	0.239 $_{.014}$	0.14 $_{.007}$
	all dom. wo. ind.	0.574 $_{.034}$	0.782 $_{.027}$	0.492 $_{.025}$	0.688 $_{.013}$	0.68 $_{.025}$	0.85 $_{.022}$	0.212 $_{.021}$	0.142 $_{.017}$
LISA	all dom. w. ind.	0.621 $_{.041}$	0.822 $_{.033}$	0.517 $_{.079}$	0.76 $_{.063}$	0.679 $_{.005}$	0.881 $_{.012}$	0.272 $_{.046}$	0.116 $_{.02}$
	all dom. wo. ind.	0.583 $_{.013}$	0.804 $_{.015}$	0.512 $_{.009}$	0.739 $_{.024}$	0.672 $_{.025}$	0.859 $_{.024}$	0.258 $_{.025}$	0.108 $_{.023}$
SWA	all dom. w. ind.	0.627 $_{.047}$	0.824 $_{.003}$	0.534 $_{.069}$	0.777 $_{.014}$	0.685 $_{.036}$	0.873 $_{.015}$	0.274 $_{.053}$	0.11 $_{.007}$
	all dom. wo. ind.	0.588 $_{.025}$	0.802 $_{.007}$	0.53 $_{.029}$	0.748 $_{.008}$	0.681 $_{.03}$	0.863 $_{.008}$	0.262 $_{.036}$	0.109 $_{.01}$

A.9.8 Qualitative Analysis: Comparison Against DRAIN and GI

To show our improved performance compared to the state-of-the-art methods DRAIN [6] and GI [7], we compare our method with the qualitative analysis performed by the authors of DRAIN on the Rotated Two Moons dataset. In this setting, all domains except the last are used for training, with the final domain reserved for testing. We illustrate our method's decision boundary compared to those provided by DRAIN in Figure 12. The accuracy results are listed in Table 7. As the contour levels are unknown, we display only the pure decision boundary for clearer comparison. Our analysis shows that our model forecasts the rotation of the two moons more accurately compared to the baselines and adapts its decision boundary more precisely.



(a) TabPFN$_{\text{dist}}$

(b) DRAIN

(c) GI

Figure 12: Comparison of our method against DRAIN [6] and GI [7] on the Rotated Two Moons dataset. The models were trained on domains $\mathcal{C} = \{0, 1, \ldots, 8\}$ and tested on domain 9. While the authors of DRAIN present different, unknown levels of the decision boundary, we present the decision boundary with 50% probability. The plots for DRAIN and GI were taken from Bai et al. [6].

Table 7: Comparison of Drift-Resilient TabPFN against DRAIN and GI on the Rotated Two Moons dataset. The metric reported is the mean out-of-distribution (OOD) accuracy along with the standard deviation. Results for DRAIN and GI are taken from Bai et al. [6].

Model	**OOD Acc.** ↑
TabPFN$_{\text{dist}}$	$\mathbf{0.98}_{.002}$
DRAIN	$0.968_{.012}$
GI	$0.965_{.014}$

A.10 Datasets

A.10.1 Validation Datasets

Synthetic Datasets

Dataset 1 (**Shifting Sin Classification**). The Shifting Sin Classification dataset is a synthetic, binary classification dataset of 1,500 instances evenly distributed across 10 domains. Each domain is differentiated by a unique shift in the offset of the sinusoid wave function, creating distinct decision boundaries for classification. The dataset contains two features corresponding to the x and y coordinates of each instance. Instances are labeled as 1 if they lie above the sine curve and 0 otherwise in their respective domains.

Dataset 2 (**Rotated Five Blobs**). The Rotated Five Blobs dataset is a synthetically generated dataset consisting of five blobs rotated sequentially counterclockwise $-20°$ around a central point in each domain. It comprises two numerical features representing the x and y coordinates of each data point. Each blob consists of 40 samples, resulting in 200 samples per domain, for a total of 2,000 samples across the 10 domains represented.

Dataset 3 (**Moving Square**). The Moving Square is a synthetically generated dataset designed for a multi-class classification task. It encompasses two features and is divided into six domains, each containing 200 instances, thereby leading to a total of 1,200 samples. In the construction of this dataset, each of the four clusters—representing distinct classes—is initially located on one corner of a square. As we transition through the six domains, each cluster progressively moves along the edge of the square to the next corner.

Dataset 4 (**Moving Diagonal Line**). The Moving Diagonal Line dataset is a synthetic dataset, generated using the sklearn blobs function. It comprises 1,200 instances, divided across 6 domains, with each domain holding 200 instances. There are two features, corresponding to the x and y coordinate of each instance. In this dataset, there are two clusters, each representing a class, following a diagonal line that moves with each domain. Thereby, both clusters move in opposite directions along parallel diagonal next to each other. Each domain in this context represents different stages of the diagonal movement.

Real World Datasets

Dataset 5 (**Indian Liver Patient Dataset**). The Indian Liver Patient Dataset (ILPD), referenced from Ramana and Venkateswarlu [51], is tailored for the binary classification task of identifying liver disease. It contains 583 records featuring 10 attributes, including age, gender, and diverse biochemical measurements. Originating from Andhra Pradesh, India, it comprises 416 liver patient records and 167 non-liver patient records. In our settings, every 5-year age interval is considered as an individual domain. The dataset, sourced from the UCI Machine Learning Repository, is geared towards supporting the diagnosis of liver disease.

Dataset 6 (**Istanbul Stock Exchange Returns**). The Istanbul Stock Exchange Returns dataset sourced from the UCI Machine Learning Repository provided by Akbilgic [52] includes 536 instances of returns from the Istanbul Stock Exchange and seven international indices from June 2009 to February 2011. The eight attributes represent various market return indices. The dataset is thereby used to predict the changes in the Istanbul stock exchange given all the other indices. The target was thereby discretized into 9 categories. The data was processed by dropping the USD column of the ISE and converting dates into a monthly domain feature, introducing a time-based distribution shift.

Dataset 7 (**Diabetes 130-US Hospitals**). The Diabetes 130-US Hospitals dataset provided by Strack et al. [53] encapsulates a decade (1999-2008) of diabetes care across 130 US hospitals, detailing 50 features related to patient demographics and hospitalization details. Criteria for inclusion are inpatient and diabetic encounters, with stays ranging from 1 to 14 days, where both lab tests and medications were administered. Features include patient identifiers, race, gender, age, admission type, duration of stay, attending physician's specialty, lab test counts, HbA1c results, diagnoses, medication details, and counts of healthcare visits prior to admission. The target of the prediction is to determine whether and in what time frame a patient will be readmitted. It is categorized into `<30 days`, `>30 days`, or `No` for no readmission. The original dataset, with 101,766 instances and 50 features, has been subsampled to 964 instances.

Dataset 8 (**Airlines Delay**). The Airlines Delay dataset, sourced from OpenML and provided by Bifet and Ikonomovska [54], contains 539,383 instances, each with 7 features. The task is to predict flight delays based on the scheduled departure information. Features include Airline, Flight, AirportFrom, AirportTo, DayOfWeek, and Time of departure. Notably, departure time, discretized to an interval of full hours, will be our distribution shift domain. The dataset was subsampled, thereby we sampled at most 60 samples per discrete time step. This resulted in 1,380 instances.

Dataset 9 (**Pima Indians Diabetes**). The Pima Indians Diabetes dataset from the National Institute of Diabetes and Digestive and Kidney Diseases, referenced by Smith et al. [55], contains 768 instances and 8 medical diagnostic features. These data represent female Pima Indian patients aged 21 or older. The task involves binary classification for predicting diabetes onset. We categorize each successive 2-year age interval as a separate domain, highlighting shifts in the dataset across age groups.

Dataset 10 (**Diabetes Prediction through Questionaire**). This dataset, collected from Sylhet Diabetes Hospital in Bangladesh and provided by Islam et al. [56], aims to predict early-stage diabetes. It comprises 520 instances and 16 features, representing symptoms and demographic information of patients. The task is binary classification, predicting whether a patient has diabetes or not. Age groups of every successive 5-year interval are considered as different domains, providing 14 age-based domains.

Dataset 11 (**Room Occupancy Detection**). The dataset is sourced from the UCI Machine Learning Repository and provided by Candanedo [57]. The preprocessed dataset, reduced to 1800 instances from the original 20,560, is used for binary classification of room occupancy based on Temperature, Humidity, Light, and CO2 levels. After removing 'date' and 'Id' features, 'day' and 'hour' were added. Thereby the 'day' was used as the temporal domain.

Dataset 12 (**Sao Paulo Urban Traffic Behavior**). The Sao Paulo Urban Traffic Behavior dataset, sourced from the UCI Machine Learning Repository provided by Ferreira et al. [58], captures records of urban traffic behavior in Sao Paulo, Brazil, from December 14 to 18, 2009, and tries to predict the slowness in traffic. The dataset contains 135 instances each with 18 attributes.

Attributes are various traffic indicators such as Hour, Immobilized bus, Broken Truck, Vehicle excess, Accident victim, and more. We have discretized the target "Slowness in traffic (%)" into intervals of 7.5 percent. Each day represents a different domain, thus introducing a time-based shift in the dataset.

A.10.2 Test Datasets

Synthetic Datasets

Dataset 13 (**Rotated Two Moons**). The Rotated Two Moons dataset as stated by Nasery et al. [7] and Bai et al. [6], is a derivative of the 2-entangled moons dataset and includes 220 instances in each of the 10 domains. Each domain is differentiated by counter-clockwise rotations of $18°$, resulting in a rotation of $18 \cdot i°$ in domain i. Also, the distribution of a subset of the instances varies across domains.

Dataset 14 (**Intersecting Blobs**). The Intersecting Blobs Dataset is a synthetically created set tailored for complex, binary classification tasks. The dataset contains two features per sample and 120 samples per domain in a total of 14 domains. Each domain comprises three classes, each represented by 40 samples appearing as blobs. These blobs move and vary, getting quite close to one another, almost intersecting. This dynamic creates sudden shifts in decision boundaries, increasing the difficulty and complexity of the classification tasks. The continuous shifts in blobs' positions across domains are implemented by adjusting their centers and standard deviations.

Dataset 15 (**Binary Label Shift**). The Binary Label Shift Dataset is a synthetic dataset tailored for binary classification in an environment with prior probability shifts. The dataset consists of 10 unique domains, with each containing 200 samples and each sample consisting of two features. The key feature of this dataset is the systematic manipulation of class probabilities across domains. This is realized by starting with a high probability of 0.95 for one class in the first domain, which progressively diminishes to 0.05 in the final domain. Simultaneously, the probability for the other class increases from 0.05 to 0.95, following the opposite direction. This dynamic essentially portrays a "fade out and fade in" pattern of the classes across the domains, representing the prior probability shift.

Dataset 16 (**Rotating Hyperplane**). The Rotating Hyperplane binary classification dataset is artificially generated based on the package `scikit-multiflow` provided by Montiel et al. [59]. It consists of five features, of which three shift over time. The dataset is divided into 15 domains of 100 instances each, providing a total of 1500 samples. As the name implies, the key aspect of this dataset is about a rotating hyperplane. In other words, the decision boundary - or hyperplane - shifts as we navigate from one domain to the next, making the classification task increasingly difficult.

Dataset 17 (**RandomRBF Drift**). The RandomRBF Drift binary classification dataset is synthetically generated based on the package `scikit-multiflow` provided by Montiel et al. [59]. This dataset has been constructed by introducing drifts in the data using Radial Basis Functions. It is characterized by the motion of cluster centroids, which are responsible for creating data drift. This movement can be visualized as clusters that change their positions, altering the distribution of data over time. The dataset consists of 8 features, 15 domains with 100 samples in each domain, for a total of 1,500 samples.

Dataset 18 (**Rotating Segments**). The Rotating Segments dataset is a synthetically generated dataset tailored for a binary classification task. The dataset visualizes a circle partitioned into four segments, similar to the slices of a cake. The data points are thereby labeled alternately. As we traverse through the ten domains, these segments undergo a rotation. Each domain contains 150 samples, accumulating to 1500 samples in total.

Dataset 19 (**Sliding Circle**). The Sliding Circle dataset is a synthetically generated dataset and represents a binary classification task. It comprises two features, the dataset is partitioned into ten domains, each possessing 200 samples, summing up to a total of 2,000 samples. The unique aspect of this dataset is its visual representation: a smaller circle slides around the inner perimeter of a larger circle. Within the larger circle, points are classified based on whether they lie inside the smaller sliding circle or outside of it. As we traverse through the ten domains, the position of the smaller circle changes, causing a shift in the classification of the points.

Dataset 20 (**Shifting Two Spirals**). The Shifting Two Spirals dataset is designed for binary classification tasks. The dataset visually represents two intertwined spirals. As we move from one domain to the next, the classification boundary in the spirals evolves. Specifically, one spiral gradually transitions its labels from the center towards the outer end, while the other spiral does the exact opposite, transitioning its labels from the outer end towards the center. This dynamic showcases a fascinating interplay of domain adaptation across the ten domains. Each domain has 200 samples, with 100 samples from each spiral, summing up to a total of 2,000 samples.

Real World Datasets

Dataset 21 (**Free Light Chain Mortality**). The free light chain dataset comprises data from a study investigating the link between serum free light chain (FLC) and mortality. It includes 1125 stratified samples per domain and target from an original pool of 7,874, featuring residents of Olmsted County, Minnesota aged 50 or more.

The task involves predicting mortality based on 9 features, which include age, sex, year of blood sample, FLC portions (kappa and lambda), the FLC group, serum creatinine, MGUS diagnosis, and days from enrollment to death or last follow-up. The feature 'chapter' was omitted because it is direct information on whether someone died or not. We treat "sample.yr", the year in which the sample was taken, as a domain, resulting in a total number of 9 domains. We suspect that both the measurements themselves and the selection of participants have changed over time. The dataset is sourced from studies by Dispenzieri et al. [60] and Kyle et al. [61].

Dataset 22 (**Electricity**). This dataset as used by Harries [62] and Gama et al. [63] contains electricity demand in the Australian New South Wales Electricity Market. Since the prices are not fixed, they fluctuate depending on supply and demand. It has 45,312 instances and 5 features. For reproducibility, the dataset includes additional features such as the New South Wales electricity price, which was used to form the target class according to the original paper, and the Victoria electricity price, which was not used in the original paper. The dataset features the demand of electricity in to provinces as well as the transfer between those for periods of 30 minutes. The task is a binary classification, which requires predicting whether the price in the current period is higher or lower than the average of the last 24 hours. The dataset contains seasonal data due to varying demand for electricity. The effects of long-term price trends on the class label are removed by the 24-hour moving average. We consider one-week periods as a single domain. To comply with TabPFN sequence length limits, we keep only two hourly intervals for each day and subsample 15 weeks of the whole time period.

Dataset 23 (**Absenteeism at Work**). The Absenteeism at Work dataset, sourced from the UCI Machine Learning Repository provided by Martiniano and Ferreira [64], comprises 740 instances across 21 features. It captures various attributes of employees and their working conditions, such as the reason for absence, day of the week, seasons, distance to work, and more, with the target feature being the absenteeism time in hours which was discredited into 4-quantiles.

The primary shift in this dataset is supposed to be seasonal. Thereby each consecutive season is treated as a different domain. Furthermore, no significant preprocessing or subsampling was required due to the manageable size of the dataset.

Dataset 24 (**Heart Disease Dataset**). The Heart Disease dataset, sourced from the UCI Machine Learning Repository and provided by Janosi et al. [65], targets the classification task of identifying the presence (values 1,2,3,4) or absence (value 0) of heart disease in patients. We treat the task as a binary classification, predicting only whether a patient has heart disease or not. It includes 303 instances, each with 13 health-related features such as age, sex, resting blood pressure, cholesterol levels, etc. In this context, each consecutive 4-year age interval is viewed as a single domain. Rows with missing values have been omitted.

Dataset 25 (**Parking Birmingham**). The Parking Birmingham dataset, provided by Stolfi [66] via the UCI Machine Learning Repository, initially comprises the capacity and occupancy rates(target) of multiple car parks. We have processed it to only include the car park labeled 'Others-CCCPS133', thereby reducing the number of instances to 1,294 from 35,717. The original 'LastUpdated' attribute has been transformed into 'day', 'week', and 'month' features, with the 'week' serving as the temporal domain. The 'Occupancy' target, originally an absolute figure, is now presented as a percentage of the parking space utilization, discretized into 25% intervals.

Dataset 26 (**Ames Housing Prices**). The Ames Housing Dataset, curated by De Cock [67], consists of 1460 instances each with 79 features detailing various aspects of residential homes in Ames, Iowa. We use only the training portion of this dataset due to the absence of ground truth targets in the test data. We have discretized the house price, which was originally a continuous variable, into a categorical variable. This transformation was achieved by partitioning the price data into intervals. Specifically, the intervals are defined as: $[0, 125k]$, $(125k, 300k]$, $(300k, \infty)$. The task is to predict the price range of a home based on its features. We treat the 'YearBuilt' attribute, divided into 15-year periods, as our domain to capture changes in housing trends over time. It is to be expected that the data set shows temporal shifts, as the price distribution between older and more modern houses differs.

Dataset 27 (**Folktables US Census**). The folktables datasets, derived from the US Census Public Use Microdata Sample (PUMS) data and published by Ding et al. [30] consists of demographic and socioeconomic data between 2015 and 2021. Each year within this timeframe represents a distinct domain for a series of tasks: ACSIncome, ACSPublicCoverage, and ACSEmployment. We purposefully limited our focus to the state Maryland, to limit the shifts to the temporal domain. The size of the datasets necessitated a stratified subsample for the target per year to reach a total of approximately 1300 instances per dataset and meet the TabPFN model requirements. This subsampling ensured a representative yet computationally manageable sample.

The tasks are as follows:

- **ACSIncome:** The task predicts whether an individual's income surpasses $50,000, narrowing the ACS PUMS data to individuals over 16 who reported at least one working hour per week in the past year and a minimum income of $100. The task consists of 10 features accross the 7 domains.
- **ACSPublicCoverage:** The objective is to predict if an individual is covered by public health insurance. The dataset is filtered to include only individuals under 65 with an income below $30,000, focusing the prediction on low-income individuals ineligible for Medicare. The task consists of 19 features accross the 7 domains.
- **ACSEmployment:** The task is to predict whether an individual is employed. The dataset is filtered to include only individuals between 16 and 90 years of age. The task consists of 16 features accross the 7 domains.

Dataset 28 (**Chess**). The Chess dataset published by Žliobaitė [68] is derived from recorded chess games, aiming to predict game outcomes (draw, lost, won) through a multi-class classification task. It consists of nine features which provide insights into the game details and player attributes, including move sequences, player side (white or black), current rating, opponent's rating, type of game, speed, and the date of the game (broken down into year, month, and day). The dataset is segmented into 27 domains, where each domain represents 20 consecutive games. This segmentation helps capture the evolution of a player's progress over time. The dataset, in its entirety, holds 533 instances. It has been constructed based on games played between 7 December 2007 and 26 March 2010.

A.11 Hyperparameter Search Spaces

Table 8: Hyperparameter search spaces we used for Wild-time baselines.

Underlying MLP (+ ERM, A-GEM, FT)	
Parameter	Values
train_update_iter	$500, 1000, 2000, 3000, 4000, 5000, 6000, 7000$
lr	$e^{\mathcal{U}(-14,-4)}$
use_scheduler	True, False
ft_scheduler_gamma	$\mathcal{U}(0.9, 1.0)$
weight_decay	$0.0, 1e-5, 1e-2$
early_stopping	True, False
early_stop_holdout	$0.1, 0.15, 0.2$
early_stop_patience	$\mathcal{U}\{10, 11, \ldots, 30\}$
LISA	
Parameter	Values
mix_alpha	$e^{\mathcal{U}(0.5,4.0)}$
cut_mix	True, False
Mixup	
Parameter	Values
mix_alpha	$e^{\mathcal{U}(-5,0)}$
EWC	
Parameter	Values
gamma	$e^{\mathcal{U}(1.0,2.0)}$
ewc_lambda	$e^{\mathcal{U}(0.5,2.0)}$
GroupDRO-T	
Parameter	Values
group_size	$\mathcal{U}\{1, 2, \ldots, 6\}$
non_overlapping	True, False
group_loss_adjustments	None, 0.1, 0.5, 1.0
group_loss_btl	True, False
SWA	
Parameter	Values
swa_portion	$\mathcal{U}(0.5, 0.9)$
swa_lr_factor	$\mathcal{U}\{1, 2, \ldots, 6\}$
IRM-T	
Parameter	Values
group_size	$\mathcal{U}\{1, 2, \ldots, 6\}$
non_overlapping	True, False
irm_lambda	$\mathcal{U}\{1, 2, \ldots, 100\}$
irm_penalty_anneal_iters	0, 250, 500, 750, 100
SI	
Parameter	Values
si_c	$\mathcal{U}(0.05, 0.2)$
epsilon	$\mathcal{U}(0.0005, 0.002)$
CORAL-T	
Parameter	Values
group_size	$\mathcal{U}\{1, 2, \ldots, 6\}$
non_overlapping	True, False
coral_lambda	$\mathcal{U}(0.1, 1.0)$

Table 9: Hyperparameter search spaces we used for our baselines.

XGBoost	
Parameter	Values
learning_rate	$e^{\mathcal{U}(-7,0)}$
max_depth	$\mathcal{U}\{1, 2, \ldots, 10\}$
subsample	$\mathcal{U}(0.2, 1)$
colsample_bytree	$\mathcal{U}(0.2, 1)$
colsample_bylevel	$\mathcal{U}(0.2, 1)$
min_child_weight	$e^{\mathcal{U}(-16,5)}$
alpha	$e^{\mathcal{U}(-16,2)}$
lambda	$e^{\mathcal{U}(-16,2)}$
gamma	$e^{\mathcal{U}(-16,2)}$
n_estimators	$\mathcal{U}\{100, 101, \ldots, 4000\}$
LightGBM	
Parameter	Values
num_leaves	$\mathcal{U}\{5, 6, \ldots, 50\}$
max_depth	$\mathcal{U}\{3, 4, \ldots, 20\}$
learning_rate	$e^{\mathcal{U}(-3,0)}$
n_estimators	$\mathcal{U}\{50, 51, \ldots, 2000\}$
min_child_weight	1e-5, 1e-3, 1e-2, 1e-1, 1, 1e1, 1e2, 1e3, 1e4
subsample	$\mathcal{U}(0.2, 0.8)$
colsample_bytree	$\mathcal{U}(0.2, 0.8)$
reg_alpha	0, 1e-1, 1, 2, 5, 7, 10, 50, 100
reg_lambda	0, 1e-1, 1, 5, 10, 20, 50, 100
CatBoost	
Parameter	Values
learning_rate	$e^{\mathcal{U}(-5,0)}$
random_strength	$\mathcal{U}\{1, 2, \ldots, 20\}$
l2_leaf_reg	$e^{\mathcal{U}(0,log(10))}$
bagging_temperature	$\mathcal{U}(0.0, 1.0)$
leaf_estimation_iterations	$\mathcal{U}\{1, 2, \ldots, 20\}$
iterations	$\mathcal{U}\{100, 101, \ldots, 4000\}$

Table 10: Preprocessing search spaces for Drift-Resilient TabPFN and TabPFN-base.

Parameter	**Search Space**
model_type	single
N_ensemble_configurations	16, None
preprocess_transforms	See Table 11
softmax_temperature	log(0.75), log(0.8), log(0.9), log(0.95)
use_poly_features	True, False
max_poly_features	50
remove_outliers	-1, 7.0, 9.0, 12.0
add_fingerprint_features	True, False
subsample_samples	0.9, 0.99, -1

Table 11: Parameters and values for enumerate_preprocess_transforms function.

Parameter	**Values**
names	["safepower"], ["quantile_uni_coarse"], ["quantile_norm_coarse"], ["adaptive"], ["norm_and_kdi"], ["quantile_uni"], ["none"], ["robust"], ["kdi_uni"], ["kdi_alpha_0.3"], ["kdi_alpha_3.0"], ["safepower", "quantile_uni"], ["kdi", "quantile_uni"], ["none", "power"]
categorical_name	["numeric", "ordinal_very_common_categories_shuffled", "onehot", "none"]
append_original	[True, False]
subsample_features	[-1, 0.99, 0.95, 0.9]
global_transformer	[None, "svd"]