

594	APPENDIX	
595		
596		
597	<b>A Experimental Setups</b>	<b>13</b>
598		
599	<b>B Baseline Methods</b>	<b>13</b>
600		
601	B.1 Machine Learning (ML) Models . . . . .	13
602	B.2 Deep Learning (Non-Foundation) Models . . . . .	14
603	B.3 Deep Learning (Foundation) Models based on ICL . . . . .	14
604		
605		
606	<b>C Baseline Implementations</b>	<b>14</b>
607		
608	<b>D Theoretical Justification</b>	<b>15</b>
609		
610	D.1 Relation to Label Shift Correction . . . . .	15
611	D.2 Bayesian Interpretation . . . . .	15
612		
613	<b>E Classical Methods for Label Shift Correction</b>	<b>16</b>
614		
615	E.1 Prior-ratio Adjustment . . . . .	16
616	E.2 EM-based Estimation . . . . .	16
617	E.3 Black-box Estimation . . . . .	16
618		
619		
620	<b>F Hyperparameter Tuning for Stronger Baselines</b>	<b>17</b>
621		
622	<b>G Dataset Statistics</b>	<b>18</b>
623		
624	<b>H K-means Clustering for Dataset Selection</b>	<b>21</b>
625		
626	<b>I Application to LoCalPFN</b>	<b>22</b>
627		
628		
629	<b>J Comparison with Methods for Label Shift Correction</b>	<b>23</b>
630		
631	<b>K Predicted Distribution of Single vs. Multiple Instances</b>	<b>24</b>
632		
633	<b>L Other Metrics</b>	<b>25</b>
634		
635		
636		
637		
638		
639		
640		
641		
642		
643		
644		
645		
646		
647		

## A EXPERIMENTAL SETUPS

**Experimental setups.** We use the official implementation of TabPFN<sup>3</sup> and adopt all default settings without modification. This includes architectural choices such as the number of layers and hidden dimensions, where we use 12 transformer layers, each with a hidden size of 192 and 6 attention heads. The feedforward layer dimension is implicitly set to 768 via a hidden factor of 4. For inference, we load the pretrained weights from TabPFN-v2<sup>4</sup> available on Hugging Face.

**Dataset.** We evaluate on 250+ tabular datasets from OpenML (Bischl et al., 2017). The dataset list is retrieved from the benchmark configuration provided in this repository<sup>5</sup>, which is built on top of the official TabPFN evaluation setup. Dataset statistics are summarized in Appendix G.

## B BASELINE METHODS

We categorize 15 baseline tabular models into three groups:

- **ML models (7):** Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (Liaw & Wiener, 2002),  $k$ -nearest neighbors ( $k$ NN), Multi-layer Perceptrons (MLP), LightGBM (Ke et al., 2017), CatBoost (Prokhorenkova et al., 2018)
- **DL (non-foundation) models (5):** FT-Transformer (Gorishniy et al., 2021), TabM (Gorishniy et al., 2024a), TabulaRNN (Thielmann & Samiee, 2024), MambaTab (Ahamed & Cheng, 2024), RealMLP (Holzmüller et al., 2024)
- **DL (foundation) models based on ICL (3):** TabPFN-v2 (Hollmann et al., 2025), LoCalPFN (Thomas et al., 2024), TabICL (Qu et al., 2025)

Details of each method are provided below.

### B.1 MACHINE LEARNING (ML) MODELS

- **Logistic Regression (LR)** (Cox, 1958): A simple linear model commonly used for binary and multiclass classification tasks in tabular data.
- **Support Vector Machine (SVM)** (Cortes & Vapnik, 1995): A kernel-based classifier that aims to find the optimal decision boundary with maximum margin between classes.
- **Multilayer Perceptron (MLP)** (Haykin, 1994): A feedforward neural network consisting of multiple fully connected layers with non-linear activations, trained via backpropagation.
- **$k$ -Nearest Neighbors ( $k$ NN)** (Altman, 1992): A non-parametric method that classifies a sample based on the majority class among its  $k$  nearest neighbors in the feature space.
- **Random Forest** (Liaw & Wiener, 2002): An ensemble learning method based on bagging over decision trees, which improves robustness and generalization.
- **LightGBM (Ke et al., 2017):** A fast and efficient GBDT model using histogram-based algorithms and leaf-wise tree growth.
- **CatBoost (Prokhorenkova et al., 2018):** A GBDT model that handles categorical features efficiently and mitigates prediction shift via ordered boosting.

<sup>3</sup><https://github.com/PriorLabs/TabPFN>

<sup>4</sup><https://huggingface.co/Prior-Labs/TabPFN-v2-clf>

<sup>5</sup>[https://github.com/carteakey/tabpfn-eval/blob/main/src/data/openml\\_list.csv](https://github.com/carteakey/tabpfn-eval/blob/main/src/data/openml_list.csv)

## B.2 DEEP LEARNING (NON-FOUNDATION) MODELS

- **FT-Transformer (Gorishniy et al., 2021)**: A transformer-based architecture tailored for tabular data, providing a simple yet powerful baseline that outperforms many prior DL models on classification and regression tasks.
- **TabM (Gorishniy et al., 2024a)**: An MLP-based model that leverages an efficient ensemble mechanism to approximate deep ensembles, enabling multiple predictions per instance while maintaining computational efficiency.
- **TabulaRNN (Thielmann & Samiee, 2024)**: An RNN-inspired architecture for tabular data that emphasizes efficiency, addressing limitations of NLP-style models in terms of scalability and training cost.
- **MambaTab (Ahamed & Cheng, 2024)**: A scalable and efficient model built on structured state-space models (SSMs), capturing long-range dependencies with fewer parameters while maintaining strong predictive performance.
- **RealMLP (Holzmüller et al., 2024)**: An enhanced MLP variant with meta-tuned hyperparameters, achieving competitive accuracy–efficiency trade-offs compared to gradient boosting methods in tabular benchmarks.

## B.3 DEEP LEARNING (FOUNDATION) MODELS BASED ON ICL

- **LoCalPFN (Thomas et al., 2024)**: A lightweight PFN variant that reduces computational cost by leveraging local task priors and architectural simplifications.
- **TabICL (Qu et al., 2025)**: A two-stage model that first applies column attention to capture feature dependencies and then row attention to encode sample interactions.
- **TabPFN-v2 (Hollmann et al., 2025)**: A state-of-the-art foundation model for tabular classification that leverages a pretrained transformer for zero-shot prediction on small datasets.

## C BASELINE IMPLEMENTATIONS

The baseline results are obtained from the following publicly available repositories:

- [1] **TabPFN Evaluation** framework<sup>6</sup> was used to evaluate all ML models, as well as the foundation models *TabPFN* and *LoCalPFN*. Since *LoCalPFN* does not have an official implementation, we reimplemented it based on the *TabPFN* codebase.
- [2] **AutoGluon v1.4.0**<sup>7</sup> was used to benchmark *TabICL* and several non-foundation models such as *FT-Transformer*, *TabM*, and *RealMLP*.
- [3] **Mambular**<sup>8</sup> provided implementations for additional non-foundation models including *MambaTab*, *TabulaRNN*, *FT-Transformer*, and *TabM*.

For models implemented in both [2] and [3] (e.g., *FT-Transformer* and *TabM*), we use the [2] versions as they yield stronger performance for a stronger baseline.

<sup>6</sup><https://github.com/carteakey/tabpfn-eval>

<sup>7</sup><https://auto.gluon.ai/>

<sup>8</sup><https://github.com/OpenTabular/DeepTabular>

## D THEORETICAL JUSTIFICATION

We provide theoretical grounding for our posterior adjustment to clarify that DistPFN is not merely a heuristic trick, but a principled approximation derived from existing theory. We present two complementary perspectives: 1) connection to classical label shift correction as a plug-in reweighting (Section D.1) and 2) Bayesian view that replaces the mismatched prior with a self-consistent estimate from model predictions (Section D.2).

### D.1 RELATION TO LABEL SHIFT CORRECTION

The label shift setting assumes that the conditional distribution  $p(x|y)$  remains invariant while the marginal priors differ:

$$p_{\text{train}}(y) \neq p_{\text{test}}(y), \quad p(x|y) \text{ is fixed.}$$

Under this assumption, the Bayes-optimal posterior is given by

$$p_{\text{test}}(y|x) \propto \frac{p_{\text{train}}(y|x)}{p_{\text{train}}(y)} p_{\text{test}}(y).$$

Classical approaches such as EM-based reweighting (Saerens et al., 2002; Lipton et al., 2018) estimate  $p_{\text{test}}(y)$  explicitly by matching marginal predictions to unlabeled test data. DistPFN instead uses the predictive marginal  $\hat{p}(y)$  obtained directly from the model, and constructs the adjustment factor

$$\alpha(y) = \frac{\hat{p}(y)}{p_{\text{train}}(y)}.$$

This yields the corrected posterior

$$\hat{p}(y|x) \propto \frac{p_{\text{train}}(y|x)}{p_{\text{train}}(y)} \hat{p}(y),$$

which can be seen as a plug-in realization of the classical correction rule, avoiding iterative estimation while remaining theoretically consistent with label shift correction.

### D.2 BAYESIAN INTERPRETATION

From a Bayesian perspective, TabPFN models the posterior under the training distribution:

$$p_{\text{train}}(y|x) \propto p(x|y) p_{\text{train}}(y).$$

At test time, however, the desired posterior is

$$p_{\text{test}}(y|x) \propto p(x|y) p_{\text{test}}(y).$$

The difference comes solely from the prior. DistPFN addresses this gap by substituting  $p_{\text{train}}(y)$  with  $\hat{p}(y)$ , the average predictive distribution obtained on the test set:

$$\hat{p}_{\text{DistPFN}}(y|x) \propto \frac{p_{\text{train}}(y|x)}{p_{\text{train}}(y)} \hat{p}(y).$$

This interpretation shows that DistPFN is not an ad-hoc adjustment but a Bayesian posterior correction where the unknown test prior is approximated in a self-consistent manner from model outputs. The method therefore inherits a principled justification while retaining the efficiency of a simple, training-free plug-in procedure.

## E CLASSICAL METHODS FOR LABEL SHIFT CORRECTION

In this section, we summarize three representative approaches for handling label shift. All of these methods directly adjust classifier outputs, but they differ in how the test prior  $\pi_{\text{test}}$  is obtained.

### E.1 PRIOR-RATIO ADJUSTMENT

Prior-ratio Adjustment (Elkan, 2001) introduces a simple correction under changing class priors in the binary setting. The method assumes that the new prior  $\pi_{\text{test}}$  is available from external knowledge or domain statistics. Given a posterior  $p_{\text{train}}(1|x)$  trained under  $\pi_{\text{train}}$ , the corrected posterior is

$$p_{\text{test}}(1|x) = \frac{p_{\text{train}}(1|x) \cdot \frac{\pi_{\text{test}}(1)}{\pi_{\text{train}}(1)}}{p_{\text{train}}(1|x) \cdot \frac{\pi_{\text{test}}(1)}{\pi_{\text{train}}(1)} + (1 - p_{\text{train}}(1|x)) \cdot \frac{1 - \pi_{\text{test}}(1)}{1 - \pi_{\text{train}}(1)}}.$$

This approach directly modifies posterior probabilities by scaling them with prior ratios. The same principle naturally extends to the multiclass case by applying the ratio  $\pi_{\text{test}}(y)/\pi_{\text{train}}(y)$  to each class posterior.

### E.2 EM-BASED ESTIMATION

EM-based Estimation (Saerens et al., 2002) proposes an iterative procedure to estimate unknown test priors when they are not directly given. At iteration  $t$ , the posterior is updated by

$$p^{(t+1)}(y|x) \propto p_{\text{train}}(y|x) \cdot \frac{\pi_{\text{test}}^{(t)}(y)}{\pi_{\text{train}}(y)}.$$

The updated posteriors provide a new estimate of  $\pi_{\text{test}}$  by averaging across the test set. Repeating this E-step and M-step allows the estimated test prior to gradually converge. The final corrected posterior then follows the standard prior-ratio adjustment, but with  $\pi_{\text{test}}$  estimated rather than assumed.

### E.3 BLACK-BOX ESTIMATION

Black-box Estimation (Lipton et al., 2018) employs a validation dataset with true labels to construct a confusion matrix  $C(s|y) = P(\hat{y} = s | y)$  that characterizes prediction errors of the classifier. On an unlabeled test set, it collects predicted labels to obtain the empirical distribution  $p_{\text{test}}(s)$ . These quantities are related through the equation

$$p_{\text{test}}(s) \approx \sum_y C(s|y) \pi_{\text{test}}(y).$$

By solving this linear system, the method estimates the test prior  $\pi_{\text{test}}$ . Once the test prior is recovered, the posterior correction is applied using the prior ratio:

$$p_{\text{test}}(y|x) \propto p_{\text{train}}(y|x) \cdot \frac{\pi_{\text{test}}(y)}{\pi_{\text{train}}(y)}.$$

This approach is considered black-box as it does not require access to classifier internals, only its predicted outputs and a validation set to estimate the confusion matrix.

## F HYPERPARAMETER TUNING FOR STRONGER BASELINES

To ensure strong and fair baselines, we perform hyperparameter tuning for each conventional ML method using the search space provided in a public implementation<sup>9</sup>. The search spaces are manually designed to cover commonly used ranges for each model class, including both optimization-related parameters (e.g., learning rate, max iterations) and regularization or structural options (e.g., penalty, tree depth, number of neighbors). We conduct random search over these spaces and tune the models on validation datasets that are kept separate from the final test splits. The details of the hyperparameter search spaces are provided in Table F.1.

Model	Hyperparameter	Type	Log-scale	Range
Logistic Regression	max_iter	int	no	{50, 100, 200, 500, 1000}
	solver	categorical	no	{newton-cg, lbfgs, liblinear, sag, saga}
	fit_intercept	boolean	no	{True, False}
	penalty	categorical	no	{l1, l2, elasticnet, none}
	C	float	no	{0.1, 1.0, 10.0, 100.0}
Random Forest	n_estimators	int	no	{10, 50, 100, 200, 500}
	criterion	categorical	no	{gini, entropy}
	probability	boolean	no	{True}
	max_depth	int / None	no	{None, 10, 50, 100, 200}
	min_samples_split	int	no	{2, 5, 10}
	min_samples_leaf	int	no	{1, 2, 4}
	max_features	categorical	no	{auto, sqrt, log2}
SVM	C	float	no	{0.1, 1.0, 10.0, 100.0}
	kernel	categorical	no	{linear, poly, rbf, sigmoid}
	probability	boolean	no	{True}
	degree	int	no	{2, 3, 4, 5}
	gamma	categorical	no	{scale, auto}
MLP	max_iter	int	no	{50, 100, 200, 500, 1000}
	activation	categorical	no	{identity, logistic, tanh, relu}
	solver	categorical	no	{lbfgs, SGD, adam}
	alpha	float	no	{0.0001, 0.001, 0.01, 0.1}
	learning_rate	categorical	no	{constant, invscaling, adaptive}
	learning_rate_init	float	no	{0.001, 0.01, 0.1}
kNN	n_neighbors	int	no	{3, 5, 11, 19}
	weights	categorical	no	{uniform, distance}
	algorithm	categorical	no	{auto, ball_tree, kd_tree, brute}
	leaf_size	int	no	{30, 50, 100}
	p	int	no	{1, 2}
XGBoost	n_estimators	int	no	{50, 100, 200}
	max_depth	int	no	{6, 10, 15, 20}
	learning_rate	float	no	{0.001, 0.01, 0.1}
	subsample	float	no	{0.5, 0.6, 0.7, 0.8, 0.9, 1.0}
	colsample_bytree	float	no	{0.4, 0.5, ..., 1.0}
	colsample_bylevel	float	no	{0.4, 0.5, ..., 1.0}
LightGBM	n_estimators	int	no	{50, 100, 200}
	max_depth	int	no	{6, 10, 15, 20}
	learning_rate	float	no	{0.001, 0.01, 0.1}
	num_leaves	int	no	{31, 60, 120, 240, 480, 960}
	min_child_samples	int	no	{10, 20, 30, 40, 50}
CatBoost	iterations	int	no	{50, 100, 200}
	depth	int	no	{6, 8, 10}
	learning_rate	float	no	{0.001, 0.01, 0.1}
	l2_leaf_reg	float	no	{1, 3, 5, 7, 9}

Table F.1: **Hyperparameter search spaces for each conventional ML baseline.** All hyperparameter values are tuned via random search over manually defined discrete sets.

<sup>9</sup><https://github.com/carteakey/tabpfn-eval>

## G DATASET STATISTICS

We evaluate on 253 tabular datasets from OpenML (Bischl et al., 2017). Summary statistics for all datasets are provided in Table G.1, G.2, and G.3. Each dataset is described using the following attributes: the dataset name (**Name**), the total number of input features (**#Features**), the number of categorical features among them (**#Cat. Feat.**), the number of data instances (**#Instances**), the number of class labels (**#Classes**), the number of missing values (**#NaNs**), and the number of samples belonging to the smallest class (**Minority Class Size**).

Name	#Features	#Cat. Feat.	#Instances	#Classes	#NaNs	Minority Class Size
pollen	6	1	3848	2	0	1924
Sick_numeric	30	1	3772	2	0	231
jungle_chess_2pcs_endgame_rat_rat	47	27	3660	2	0	1605
UCI_churn	21	1	3333	2	0	483
led24	25	25	3200	10	0	296
led7	8	8	3200	10	0	270
kr-vs-kp	37	37	3196	2	0	1527
splice	61	61	3190	3	0	767
space_ga	7	1	3107	2	0	1541
StackOverflow-polarity-train	2	1	3097	3	0	842
seismic-bumps	19	5	2584	2	0	170
ozone-level-8hr	73	1	2534	2	0	160
jungle_chess_2pcs_endgame_lion_lion	47	27	2352	2	0	949
jungle_chess_2pcs_endgame_elephant_elephant	47	27	2351	2	0	1035
segment	20	1	2310	7	0	330
Titanic	4	1	2201	2	0	711
quake	4	1	2178	2	0	969
kc1	22	1	2109	2	0	326
balloon	2	1	2001	2	0	482
mfeat-fourier	77	1	2000	10	0	200
ozone-level-8hr_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	73	1	2000	2	0	126
mfeat_karluenen	65	1	2000	10	0	200
jannis_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	55	1	2000	2	0	1000
coverttype_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	55	45	2000	2	0	1000
first-order-theorem-proving_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	52	1	2000	6	0	159
MiniBooNE_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	51	1	2000	2	0	1000
KDDCup09_upselling_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	50	16	2000	2	0	1000
ada_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	49	1	2000	2	0	496
mfeat-zernike	48	1	2000	10	0	200
connect-4_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	43	43	2000	3	0	191
kr-vs-kp_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	37	37	2000	2	0	956
road-safety_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	33	4	2000	2	0	1000
GesturePhaseSegmentationProcessed_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	33	1	2000	5	0	202
PhishingWebsites_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	31	31	2000	2	0	886
pol_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	27	1	2000	2	0	1000
Higgs_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	25	1	2000	2	0	1000
eye_movements_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	24	4	2000	2	0	1000
numera128.6_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	22	1	2000	2	0	990
kc1_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	22	1	2000	2	0	309
kdd_ipums_la_97_small_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	21	1	2000	2	0	1000
churn_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	21	5	2000	2	0	283
compass_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	18	10	2000	2	0	1000
house_16H_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	17	1	2000	2	0	1000
segment_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	17	1	2000	7	0	285
adult_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	15	9	2000	2	242	479
adult_seed_1_nrows_2000.nclasses_10.ncols_100.stratify.True	15	9	2000	2	248	479
adult_seed_2_nrows_2000.nclasses_10.ncols_100.stratify.True	15	9	2000	2	279	479
adult_seed_3_nrows_2000.nclasses_10.ncols_100.stratify.True	15	9	2000	2	254	479
adult_seed_4_nrows_2000.nclasses_10.ncols_100.stratify.True	15	9	2000	2	253	479
rl_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	13	8	2000	2	0	1000
wine_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	12	1	2000	2	0	1000
Click_prediction_small_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	12	7	2000	2	0	337
Amazon_employee_access_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	10	10	2000	2	0	116
california_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	9	1	2000	2	0	1000
sf-police-incidents_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	9	6	2000	2	0	243
electricity_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	8	1	2000	2	0	1000
airlines_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	8	5	2000	2	0	891
mfeat-morphological	7	1	2000	10	0	200
jungle_chess_2pcs_raw_endgame_complete_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	7	1	2000	3	0	194
phoneme_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	6	1	2000	2	0	1000
wilt_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	6	1	2000	2	0	108
steel-plates-fault	34	1	1941	2	0	673
steel-plates-fault_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	28	1	1941	7	0	55
GAMETES_Epistasis_2-Way_20atts_0.1H_EDM-1-1	21	21	1600	2	0	800
pc3	38	1	1563	2	0	160
cmc	10	8	1473	3	0	333
cmc_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	10	8	1473	3	0	333
ibm-employee-performance	34	1	1470	2	0	226
pc4	38	1	1458	2	0	178
pc4_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	38	1	1458	2	0	178
banknote-authentication	5	1	1372	2	0	610
analcadata_halloffame	17	2	1340	2	20	125
mofn-3-7-10	11	11	1324	2	0	292
socmob	6	5	1156	2	0	256
parity5_plus_5	11	11	1124	2	0	557
PieChart3	38	1	1077	2	0	134
qsar-biodeg	42	1	1055	2	0	356
qsar-biodeg_seed_0_nrows_2000.nclasses_10.ncols_100.stratify.True	42	1	1055	2	0	356
PizzaCutter3	38	1	1043	2	0	127
rmfisa_sleepdata	3	1	1024	4	0	94
credit-g	21	14	1000	2	0	300
dummy	7	1	1000	2	0	273
xd6	10	10	973	2	0	322
tokyo1	45	3	959	2	0	346
tic-tac-toe	10	10	958	2	0	332
Tour-and-Travels-Customer-Churn-Prediction	7	5	954	2	60	224
stock	10	1	950	2	0	462
vehicle	19	1	846	4	0	199
vehicle_reproduced	19	1	846	4	0	199
analcadata_authorship	71	1	841	4	0	55

Table G.1: Dataset statistics - Part 1

Name	#Features	#Cat. Feat.	#Instances	#Classes	#NaNs	Minority Class Size
analcatdata_dmft	5	5	797	6	0	123
diabetes	9	1	768	2	0	268
blood-transfusion-service-center	5	1	748	2	0	178
blood-transfusion-service-center_seed_0_nrows_2000_nclasses_10_ncols_100_stratify_True	5	1	748	2	0	178
doa_bwin_balanced	14	3	708	2	0	354
PieChart1	38	1	705	2	0	61
breast-w	10	1	699	2	16	241
credit-approval	16	10	690	2	67	307
credit-approval_reproduced	16	10	690	2	67	307
Australian	15	9	690	2	0	307
Australian_seed_0_nrows_2000_nclasses_10_ncols_100_stratify_True	15	9	690	2	0	307
disclosure_x_bias	4	1	662	2	0	317
disclosure_x_tampered	4	1	662	2	0	327
disclosure_x_noise	4	1	662	2	0	329
disclosure_z	4	1	662	2	0	314
PizzaCutter1	38	1	661	2	0	52
balance-scale	5	1	625	3	0	49
monks-problems-2	7	7	601	2	0	206
monks-problems-1	61	1	600	6	0	100
synthetic_control	12	12	576	2	0	239
sensor	31	1	569	2	0	212
wdbc	5	2	559	2	0	80
arsenic-female-bladder	7	7	556	2	0	278
monks-problems-3	7	7	554	2	0	266
climate-model-simulation-crashes	21	1	540	2	0	46
climate-model-simulation-crashes	14	3	530	2	0	176
CPMP-2015-runtime-classification	23	1	527	4	0	78
kc2	22	1	522	2	0	107
threeOf9	10	10	512	2	0	238
rmftsa_ladata	11	1	508	2	0	222
boston_corrected	21	4	506	2	0	223
boston	14	2	506	2	0	209
collins	23	3	500	2	0	80
pm10	8	1	500	2	0	246
no2	8	1	500	2	0	249
LED-display-domain-7digit	8	1	500	10	0	37
irish	6	4	500	2	32	222
PopularKids	11	5	478	3	0	90
analcatdata_apnea2	4	3	475	2	0	64
analcatdata_apnea1	4	3	475	2	0	61
thoracic-surgery	17	14	470	2	0	70
analcatdata_vineyard	4	2	468	2	0	208
chscase_vine2	3	1	468	2	0	212
sa-heart	10	2	462	2	0	160
analcatdata_apnea3	4	3	450	2	0	55
wholesale-customers	9	2	440	2	0	142
mw1	38	1	403	2	0	31
user-knowledge	6	1	403	5	0	24
chscase_census5	8	1	400	2	0	193
chscase_census4	8	1	400	2	0	194
chscase_census3	8	1	400	2	0	192
chscase_census2	8	1	400	2	0	197
chscase_census6	7	1	400	2	0	165
analcatdata_germangss	6	5	400	4	0	100
calendarDOW	33	21	399	5	0	44
autoMpg	8	4	398	2	6	189
vinnie	3	1	380	2	0	185
jEdit_4.2.4.3	9	1	369	2	0	165
dermatology	35	34	366	6	8	20
analcatdata_draft	5	3	366	2	1	32
analcatdata_birthday	4	3	365	2	30	53
ionosphere	35	1	351	2	0	126
SPECTF	45	1	349	2	0	95
penguins	7	3	344	3	18	68
CastMetal1	38	1	327	2	0	42
visualizing-galaxy	5	1	323	2	0	148
plasma_retinol	14	4	315	2	0	133
solar-flare	13	13	315	5	0	21
diggle_table_a2	9	1	310	9	0	18
vertebra-column	7	1	310	3	0	60
haberman	4	2	306	2	0	81
heart-c	14	8	303	2	7	138
cleveland	14	8	303	2	6	139
cholesterol	14	8	303	2	6	137
cleve	14	9	303	2	0	138
cleveland-nominal	8	8	303	5	0	13
CostaMadr1	38	1	296	2	0	38
Heart_disease_prediction_20	14	1	296	2	0	137
breast-cancer	10	10	286	2	9	85
breastTumor	10	9	286	2	9	120
analcatdata_broadwaymult	8	5	285	7	27	21
mu284	11	1	284	2	0	142
DiabeticMellitus	98	1	281	2	2	99
breast-cancer-dropped-missing-attributes-values	10	10	277	2	0	81
jEdit_4.0.4.2	9	1	274	2	0	134
heart-statlog	14	1	270	2	0	120
SPECT	23	23	267	2	0	55
Touch2	11	1	265	8	0	27
analcatdata_lawsuit	5	2	264	2	0	19
rmftsa_ctoarrivals	3	2	264	2	0	101

Table G.2: Dataset statistics - Part 2



Name	#Features	#Cat. Feat.	#Instances	#Classes	#NaNs	Minority Class Size
MegaWatt1	38	1	253	2	0	27
bodyfat	15	1	252	2	0	124
qualitative-bankruptcy	7	7	250	2	0	107
prnn_synth	3	1	250	2	0	125
conference.attendance	7	7	246	2	0	31
chatfield.4	13	1	235	2	0	93
chscase_whale	9	1	228	2	20	111
lungcancer.GSE31210	24	3	226	2	0	35
chscase_geyser1	3	1	222	2	0	88
thyroid-new	6	1	215	3	0	30
glass	10	1	214	6	0	9
prnn_fglass	10	1	214	2	0	76
seeds	8	1	210	3	0	70
biomed	9	2	209	2	15	75
cpu	8	2	209	2	0	53
machine_cpu	7	1	209	2	0	56
sonar	61	1	208	2	0	97
regime_alimentaire	20	17	202	2	17	41
heart-long-beach	14	1	200	5	0	10
pwLinear	11	1	200	2	0	97
prnn_crabs	8	2	200	2	0	100
parkinsons	23	1	195	2	0	48
pharynx	11	10	195	2	2	74
KnuggetChase3	40	1	194	2	0	36
wisconsin	33	1	194	2	0	90
lowbwt	10	8	189	2	0	90
triazines	61	1	186	2	0	77
chscase_funds	3	1	185	2	0	87
planning-relax	13	1	182	2	0	52
Smartphone-Based_Recognition_of_Human_Activities	68	2	180	6	0	30
backache	32	27	180	2	0	25
wine	14	1	178	3	0	48
servo	5	5	167	2	0	38
robot-failures-lp5	91	1	164	5	0	21
analcdata_wildcat	6	3	163	2	0	47
mc2	40	1	161	2	0	52
corral	7	7	160	2	0	70
hayes-roth	5	1	160	3	0	31
auto_price	16	2	159	2	0	54
autoPrice	16	1	159	2	0	54
analcdata_gsssexsurvey	10	6	159	2	6	35
TuningSVMs	81	1	156	2	0	54
grub-damage	9	7	155	4	0	19
teachingAssistant	7	5	151	3	0	49
tae	6	3	151	3	0	49
iris	5	1	150	3	0	50
iris-example	5	1	150	3	0	50
sleuth_case2002	7	5	147	2	0	69
kc1-top5	95	1	145	2	0	8
kc1-binary	95	1	145	2	0	60
newton_hema	4	2	140	2	0	70
veteran	8	5	137	2	0	43
analcdata_boxing2	4	4	132	2	0	61
analcdata_seropositive	4	2	132	2	0	46
transplant	4	1	131	2	0	48
datatrive	9	1	130	2	0	11
visualizing_livestock	3	2	130	5	0	26
humandevl	2	1	130	2	0	65
mux6	7	7	128	2	0	64
MindCave2	40	1	125	2	0	44
fruitfly	5	3	125	2	0	49
KungChi3	40	1	123	2	0	16
heart-switzerland	13	1	123	5	0	5
ar1	30	1	121	2	0	9
analcdata_boxing1	4	4	120	2	0	42
rabe_266	3	1	120	2	0	57
robot-failures-lp4	91	1	117	3	0	21
visualizing_environmental	4	1	111	2	0	53
cloud	8	2	108	2	0	32
analcdata_michiganacc	4	3	108	2	0	48
ar4	30	1	107	2	0	20
molecular-biology_promoters	58	58	106	2	0	53
breast-tissue	10	1	106	6	0	14
ar6	30	1	101	2	0	15
zoo	17	16	101	7	0	4
fertility	10	1	100	2	0	12
analcdata_creditscore	7	4	100	2	0	27
blogger	6	6	100	2	0	32
analcdata_chlamydia	4	4	100	2	0	19
analcdata_neavote	3	2	100	2	0	7

Table G.3: Dataset statistics - Part 3

## H K-MEANS CLUSTERING FOR DATASET SELECTION

Table H.1 reports the extended results of our K-means clustering-based training set selection under different numbers of clusters  $K \in \{3, 5, 10\}$ , where a proportion ( $P \in \{0.05, 0.10, 0.20\}$ ) of samples is drawn from each cluster. Across all settings, our method demonstrates stable performance regardless of  $K$ , confirming its robustness when applied with clustering-based selection.

$K$	$P$	Methods	w/o shift	Shift strength ( $\beta$ )						
				0.0	0.1	0.5	1.0	2.0	5.0	Avg.
3	0.05	TabPFN-v2	0.668	0.596	0.591	0.548	0.500	0.439	0.408	0.513
		DistPFN	0.661	<u>0.622</u>	<u>0.614</u>	<u>0.579</u>	<u>0.532</u>	<u>0.454</u>	<u>0.428</u>	<u>0.538</u>
		DistPFN-T	0.657	<b>0.625</b>	<b>0.616</b>	<b>0.588</b>	<b>0.540</b>	<b>0.459</b>	<b>0.433</b>	<b>0.543</b>
	0.10	TabPFN-v2	0.699	0.626	0.632	0.570	0.528	0.465	0.424	0.541
		DistPFN	0.692	<u>0.641</u>	<u>0.653</u>	<u>0.620</u>	<u>0.564</u>	<u>0.498</u>	<u>0.450</u>	<u>0.573</u>
		DistPFN-T	0.687	<b>0.643</b>	<b>0.657</b>	<b>0.628</b>	<b>0.569</b>	<b>0.504</b>	<b>0.454</b>	<b>0.584</b>
	0.20	TabPFN-v2	0.732	0.673	0.668	0.626	0.576	0.505	0.468	0.591
		DistPFN	0.727	<u>0.692</u>	<u>0.688</u>	<u>0.669</u>	<u>0.614</u>	<u>0.556</u>	<u>0.509</u>	<u>0.639</u>
		DistPFN-T	0.722	<b>0.691</b>	<b>0.692</b>	<b>0.674</b>	<b>0.620</b>	<b>0.568</b>	<b>0.516</b>	<b>0.661</b>
5	0.05	TabPFN-v2	0.676	0.605	0.606	0.550	0.503	0.459	0.429	0.529
		DistPFN	0.673	<u>0.628</u>	<u>0.630</u>	<u>0.587</u>	<u>0.534</u>	<u>0.487</u>	<u>0.453</u>	<u>0.561</u>
		DistPFN-T	0.672	<b>0.629</b>	<b>0.634</b>	<b>0.594</b>	<b>0.539</b>	<b>0.493</b>	<b>0.460</b>	<b>0.565</b>
	0.10	TabPFN-v2	0.699	0.631	0.644	0.586	0.540	0.485	0.446	0.569
		DistPFN	0.696	<u>0.654</u>	<u>0.665</u>	<u>0.624</u>	<u>0.583</u>	<u>0.528</u>	<u>0.475</u>	<u>0.609</u>
		DistPFN-T	0.693	<b>0.655</b>	<b>0.670</b>	<b>0.630</b>	<b>0.591</b>	<b>0.538</b>	<b>0.483</b>	<b>0.620</b>
	0.20	TabPFN-v2	0.732	0.670	0.679	0.628	0.582	0.523	0.481	0.618
		DistPFN	0.736	<u>0.687</u>	<u>0.697</u>	<u>0.662</u>	<u>0.625</u>	<u>0.576</u>	<u>0.521</u>	<u>0.645</u>
		DistPFN-T	0.731	<b>0.690</b>	<b>0.698</b>	<b>0.670</b>	<b>0.631</b>	<b>0.584</b>	<b>0.531</b>	<b>0.667</b>
10	0.05	TabPFN-v2	0.708	0.644	0.639	0.591	0.547	0.505	0.460	0.554
		DistPFN	0.706	<u>0.659</u>	<u>0.662</u>	<u>0.627</u>	<u>0.586</u>	<u>0.548</u>	<u>0.493</u>	<u>0.589</u>
		DistPFN-T	0.701	<b>0.662</b>	<b>0.667</b>	<b>0.640</b>	<b>0.601</b>	<b>0.562</b>	<b>0.504</b>	<b>0.605</b>
	0.10	TabPFN-v2	0.727	0.663	0.664	0.617	0.582	0.534	0.481	0.620
		DistPFN	0.723	<u>0.679</u>	<u>0.685</u>	<u>0.650</u>	<u>0.618</u>	<u>0.577</u>	<u>0.515</u>	<u>0.642</u>
		DistPFN-T	0.718	<b>0.685</b>	<b>0.691</b>	<b>0.656</b>	<b>0.629</b>	<b>0.588</b>	<b>0.527</b>	<b>0.652</b>
	0.20	TabPFN-v2	0.749	0.697	0.689	0.651	0.619	0.561	0.510	0.638
		DistPFN	0.749	<u>0.713</u>	<u>0.711</u>	<u>0.682</u>	<u>0.663</u>	<u>0.610</u>	<u>0.553</u>	<u>0.676</u>
		DistPFN-T	0.748	<b>0.716</b>	<b>0.715</b>	<b>0.689</b>	<b>0.670</b>	<b>0.620</b>	<b>0.563</b>	<b>0.688</b>

Table H.1: **K-Means-based training dataset selection.** Our method remains effective when training subsets are selected by clustering the data and sampling a percentage ( $P$ ) of samples from each of  $K$  clusters.

## I APPLICATION TO LoCALPFN

Table I.1 provides the full results for LoCalPFN under different values of  $k$  across six  $\beta$  values. The results confirm that our methods yield consistent improvements regardless of the choice of  $k$ , demonstrating robustness of the approach.

$k$	Methods	Shift strength ( $\beta$ )						Avg.
		0.0	0.1	0.5	1.0	2.0	5.0	
3	LoCalPFN	0.789	0.787	0.774	0.758	0.711	0.679	0.750
	+ DistPFN	<u>0.794</u>	<u>0.794</u>	<u>0.792</u>	<u>0.786</u>	<u>0.772</u>	<u>0.752</u>	<u>0.782</u>
	+ DistPFN-T	<b>0.794</b>	<b>0.794</b>	<b>0.794</b>	<b>0.790</b>	<b>0.779</b>	<b>0.759</b>	<b>0.785</b>
5	LoCalPFN	0.792	0.791	0.785	0.775	0.744	0.714	0.767
	+ DistPFN	<u>0.794</u>	<u>0.795</u>	<u>0.793</u>	<u>0.790</u>	<u>0.777</u>	<u>0.766</u>	<u>0.786</u>
	+ DistPFN-T	<b>0.795</b>	<b>0.796</b>	<b>0.795</b>	<b>0.794</b>	<b>0.784</b>	<b>0.770</b>	<b>0.789</b>
10	LoCalPFN	0.794	0.792	0.786	0.778	0.752	0.720	0.770
	+ DistPFN	<u>0.796</u>	<u>0.795</u>	<u>0.793</u>	<u>0.791</u>	<u>0.779</u>	<u>0.768</u>	<u>0.787</u>
	+ DistPFN-T	<b>0.797</b>	<b>0.797</b>	<b>0.796</b>	<b>0.794</b>	<b>0.785</b>	<b>0.774</b>	<b>0.789</b>
20	LoCalPFN	0.794	0.793	0.788	0.778	0.753	0.719	0.771
	+ DistPFN	<u>0.797</u>	<u>0.796</u>	<u>0.794</u>	<u>0.790</u>	<u>0.782</u>	<u>0.770</u>	<u>0.788</u>
	+ DistPFN-T	<b>0.798</b>	<b>0.797</b>	<b>0.796</b>	<b>0.794</b>	<b>0.787</b>	<b>0.776</b>	<b>0.791</b>

Table I.1: **Application to LoCalPFN.** DistPFN and DistPFN-T applied to LoCalPFN show consistent improvements across varying numbers of neighbors ( $k$ ).

## J COMPARISON WITH METHODS FOR LABEL SHIFT CORRECTION

To demonstrate the effectiveness of our approach, we compare it with classical methods for handling label shift by rescaling classifier outputs, which typically require estimating the test distribution: EM-based Estimation (EME) (Saerens et al., 2002) and Black-box Estimation (BBE) (Lipton et al., 2018). Table J.1 presents the results, showing that our method is effective without requiring estimation of the test prior.

Methods	w/o shift	Shift strength ( $\beta$ )						Avg.
		0.0	0.1	0.5	1.0	2.0	5.0	
LoCalPFN	<b>0.816</b>	0.794	0.793	0.788	0.778	0.753	0.719	0.771
+ EME	0.801	0.792	0.790	0.786	0.785	0.778	0.769	0.783
+ BBE	0.805	<b>0.798</b>	0.795	0.792	0.789	<u>0.782</u>	<u>0.770</u>	0.787
+ DistPFN	<b>0.816</b>	<u>0.797</u>	<u>0.796</u>	<u>0.794</u>	0.790	<u>0.782</u>	<u>0.770</u>	<u>0.788</u>
+ DistPFN-T	<b>0.816</b>	<b>0.798</b>	<b>0.797</b>	<b>0.796</b>	<b>0.794</b>	<b>0.787</b>	<b>0.776</b>	<b>0.791</b>
TabICL	<b>0.806</b>	0.783	0.781	0.770	0.747	0.704	0.664	0.742
+ EME	0.798	0.776	0.776	0.770	0.769	0.761	0.747	0.766
+ BBE	0.802	0.783	0.785	0.780	0.774	0.754	0.734	0.768
+ DistPFN	<b>0.806</b>	<u>0.786</u>	<u>0.786</u>	<u>0.781</u>	<u>0.776</u>	<u>0.763</u>	<u>0.746</u>	<u>0.773</u>
+ DistPFN-T	<b>0.806</b>	<b>0.786</b>	<b>0.786</b>	<b>0.783</b>	<b>0.780</b>	<b>0.771</b>	<b>0.755</b>	<b>0.777</b>
TabPFN-v2	<b>0.818</b>	<u>0.797</u>	0.796	0.790	0.782	0.759	0.727	0.775
+ EME	0.801	0.793	0.793	0.790	0.787	<u>0.783</u>	0.768	0.786
+ BBE	0.805	<b>0.799</b>	<u>0.797</u>	<b>0.797</b>	<u>0.791</u>	<u>0.783</u>	0.768	<u>0.789</u>
+ DistPFN	<b>0.818</b>	<b>0.799</b>	<u>0.797</u>	<u>0.795</u>	<u>0.791</u>	<u>0.783</u>	<u>0.769</u>	<u>0.789</u>
+ DistPFN-T	<b>0.818</b>	<b>0.799</b>	<b>0.798</b>	<b>0.797</b>	<b>0.796</b>	<b>0.789</b>	<b>0.775</b>	<b>0.792</b>

Figure J.1: Comparison with other label shift methods.

## K PREDICTED DISTRIBUTION OF SINGLE VS. MULTIPLE INSTANCES

As TabPFN produces identical predictions whether test instances are evaluated individually or in batches, DistPFN and DistPFN-T can adjust based on either 1) the prediction of a *single* instance or 2) the average prediction across *multiple* instances. As shown in Table K.1, both choices consistently improve TabPFN-v2 (Hollmann et al., 2025), averaged across six  $\beta$ s for *w/ shift*, demonstrating robustness to the choice of distribution source.

	Pred. distn.	w/o shift	Shift strength ( $\beta$ )						
			0.0	0.1	0.5	1.0	2.0	5.0	Avg.
TabPFN-v2	-	0.818	0.797	0.796	0.790	0.782	0.759	0.727	0.775
+ DistPFN	Single	<b>0.818</b>	<u>0.797</u>	<u>0.796</u>	<b>0.795</b>	<b>0.793</b>	<b>0.784</b>	<b>0.770</b>	<b>0.789</b>
	Multiple	<b>0.818</b>	<u>0.799</u>	<u>0.797</u>	<b>0.795</b>	<u>0.791</u>	<u>0.783</u>	<b>0.770</b>	<b>0.789</b>
+ DistPFN-T	Single	<b>0.818</b>	<u>0.797</u>	<u>0.797</u>	<u>0.796</u>	<u>0.795</u>	<u>0.788</u>	<u>0.773</u>	<u>0.791</u>
	Multiple	<b>0.818</b>	<u>0.799</u>	<u>0.798</u>	<u>0.797</u>	<u>0.796</u>	<u>0.789</u>	<u>0.775</u>	<u>0.792</u>

Table K.1: **Predicted distributions: Single vs. Multiple.** The proposed methods consistently improves TabPFN-v2 regardless of whether the adjustment is based on single or aggregated distribution.

## L OTHER METRICS

Table L.1 reports the comparison of our methods and baselines under  $\beta = 2$  in terms of ROC-AUC, demonstrating the effectiveness of our method. The results demonstrate that our method shows nearly the same values as the backbone, as the adjustment only rescales predicted probabilities without altering their order.

Methods		w/o shift	Shift strength ( $\beta$ )							
			0.0	0.1	0.5	1.0	2.0	5.0	Avg.	
Machine Learning	LogReg. + HPO	0.813 $\pm$ 0.002 0.817 $\pm$ 0.002	0.789 $\pm$ 0.002 0.806 $\pm$ 0.001	0.789 $\pm$ 0.002 0.806 $\pm$ 0.001	0.790 $\pm$ 0.002 0.805 $\pm$ 0.002	0.788 $\pm$ 0.002 0.803 $\pm$ 0.002	0.784 $\pm$ 0.003 0.797 $\pm$ 0.001	0.777 $\pm$ 0.002 0.791 $\pm$ 0.001	0.786 0.801	
	SVM + HPO	0.815 $\pm$ 0.002 0.840 $\pm$ 0.002	0.744 $\pm$ 0.004 0.804 $\pm$ 0.003	0.747 $\pm$ 0.005 0.804 $\pm$ 0.003	0.750 $\pm$ 0.004 0.800 $\pm$ 0.001	0.744 $\pm$ 0.004 0.799 $\pm$ 0.004	0.733 $\pm$ 0.005 0.794 $\pm$ 0.002	0.725 $\pm$ 0.006 0.785 $\pm$ 0.002	0.741 0.798	
	MLP + HPO	0.821 $\pm$ 0.003 0.849 $\pm$ 0.003	0.747 $\pm$ 0.002 0.799 $\pm$ 0.001	0.750 $\pm$ 0.003 0.799 $\pm$ 0.002	0.746 $\pm$ 0.005 0.796 $\pm$ 0.001	0.735 $\pm$ 0.004 0.788 $\pm$ 0.003	0.719 $\pm$ 0.002 0.781 $\pm$ 0.003	0.702 $\pm$ 0.004 0.772 $\pm$ 0.002	0.733 0.789	
	kNN + HPO	0.789 $\pm$ 0.001 0.828 $\pm$ 0.002	0.728 $\pm$ 0.003 0.775 $\pm$ 0.002	0.728 $\pm$ 0.004 0.775 $\pm$ 0.003	0.727 $\pm$ 0.004 0.773 $\pm$ 0.002	0.722 $\pm$ 0.004 0.768 $\pm$ 0.002	0.709 $\pm$ 0.003 0.756 $\pm$ 0.001	0.693 $\pm$ 0.003 0.742 $\pm$ 0.002	0.718 0.765	
	Random Forest + HPO	0.836 $\pm$ 0.003 0.849 $\pm$ 0.003	0.824 $\pm$ 0.003 0.836 $\pm$ 0.003	0.823 $\pm$ 0.003 0.835 $\pm$ 0.003	0.821 $\pm$ 0.003 0.834 $\pm$ 0.003	0.818 $\pm$ 0.002 0.832 $\pm$ 0.002	0.812 $\pm$ 0.001 0.826 $\pm$ 0.001	0.802 $\pm$ 0.001 0.818 $\pm$ 0.002	0.817 0.830	
	LightGBM + HPO	0.824 $\pm$ 0.002 0.845 $\pm$ 0.003	0.813 $\pm$ 0.001 0.776 $\pm$ 0.009	0.812 $\pm$ 0.001 0.767 $\pm$ 0.006	0.809 $\pm$ 0.002 0.781 $\pm$ 0.003	0.805 $\pm$ 0.003 0.774 $\pm$ 0.010	0.797 $\pm$ 0.002 0.775 $\pm$ 0.009	0.785 $\pm$ 0.003 0.779 $\pm$ 0.004	0.805 0.775	
	CatBoost + HPO	0.847 $\pm$ 0.003 0.843 $\pm$ 0.003	0.833 $\pm$ 0.003 0.833 $\pm$ 0.003	0.832 $\pm$ 0.002 0.832 $\pm$ 0.002	0.830 $\pm$ 0.003 0.830 $\pm$ 0.003	0.827 $\pm$ 0.002 0.827 $\pm$ 0.002	0.820 $\pm$ 0.002 0.819 $\pm$ 0.002	0.810 $\pm$ 0.002 0.810 $\pm$ 0.001	0.825 0.825	
	Non-found.	FT-Transformer	0.821 $\pm$ 0.003	0.818 $\pm$ 0.003	0.819 $\pm$ 0.002	0.816 $\pm$ 0.003	0.812 $\pm$ 0.002	0.795 $\pm$ 0.003	0.771 $\pm$ 0.002	0.805
		TabM	0.824 $\pm$ 0.003	0.824 $\pm$ 0.003	0.824 $\pm$ 0.002	0.823 $\pm$ 0.003	0.821 $\pm$ 0.001	0.808 $\pm$ 0.003	0.791 $\pm$ 0.002	0.815
		TabulaRNN	0.774 $\pm$ 0.003	0.699 $\pm$ 0.003	0.684 $\pm$ 0.003	0.641 $\pm$ 0.004	0.585 $\pm$ 0.009	0.522 $\pm$ 0.011	0.465 $\pm$ 0.008	0.599
MambaTab		0.743 $\pm$ 0.005	0.629 $\pm$ 0.006	0.603 $\pm$ 0.004	0.525 $\pm$ 0.002	0.466 $\pm$ 0.010	0.430 $\pm$ 0.005	0.394 $\pm$ 0.002	0.508	
RealMLP		0.821 $\pm$ 0.002	0.805 $\pm$ 0.003	0.806 $\pm$ 0.002	0.807 $\pm$ 0.002	0.804 $\pm$ 0.003	0.795 $\pm$ 0.000	0.781 $\pm$ 0.004	0.800	
Foundation	LoCalPFN + DistPFN + DistPFN-T	0.858 $\pm$ 0.002 0.858 $\pm$ 0.002 0.858 $\pm$ 0.002	0.842 $\pm$ 0.002 0.842 $\pm$ 0.002 0.842 $\pm$ 0.002	0.840 $\pm$ 0.001 0.840 $\pm$ 0.002 0.840 $\pm$ 0.002	0.839 $\pm$ 0.000 0.839 $\pm$ 0.001 0.839 $\pm$ 0.001	0.836 $\pm$ 0.000 0.836 $\pm$ 0.001 0.837 $\pm$ 0.001	0.830 $\pm$ 0.000 0.830 $\pm$ 0.001 0.830 $\pm$ 0.001	0.826 $\pm$ 0.001 0.826 $\pm$ 0.002 0.826 $\pm$ 0.003	0.836 0.836 0.836	
	TabICL + DistPFN + DistPFN-T	0.845 $\pm$ 0.003 0.845 $\pm$ 0.003 0.845 $\pm$ 0.003	0.832 $\pm$ 0.003 0.832 $\pm$ 0.003 0.832 $\pm$ 0.003	0.832 $\pm$ 0.001 0.832 $\pm$ 0.001 0.832 $\pm$ 0.001	0.830 $\pm$ 0.002 0.830 $\pm$ 0.002 0.830 $\pm$ 0.002	0.826 $\pm$ 0.002 0.826 $\pm$ 0.002 0.826 $\pm$ 0.002	0.821 $\pm$ 0.002 0.821 $\pm$ 0.002 0.821 $\pm$ 0.002	0.813 $\pm$ 0.003 0.814 $\pm$ 0.003 0.814 $\pm$ 0.003	0.826 0.826 0.826	
	TabPFN-v2 + DistPFN + DistPFN-T	0.859 $\pm$ 0.002 0.859 $\pm$ 0.002 0.859 $\pm$ 0.002	0.843 $\pm$ 0.002 0.843 $\pm$ 0.002 0.843 $\pm$ 0.002	0.842 $\pm$ 0.003 0.843 $\pm$ 0.001 0.842 $\pm$ 0.003	0.841 $\pm$ 0.002 0.841 $\pm$ 0.002 0.841 $\pm$ 0.002	0.838 $\pm$ 0.002 0.838 $\pm$ 0.002 0.838 $\pm$ 0.002	0.833 $\pm$ 0.001 0.833 $\pm$ 0.001 0.833 $\pm$ 0.001	0.826 $\pm$ 0.002 0.826 $\pm$ 0.003 0.826 $\pm$ 0.003	0.837 0.837 0.837	

Table L.1: Tabular classification results: ROC-AUC comparisons.