A Variational Estimator for L_p Calibration Errors

Eugène Berta[*,1,2] **Sacha Braun**[*,1,2] **David Holzmüller**[*,1] **Michael I. Jordan**[1,4] **Francis Bach**[1,2]
[1]INRIA [2]Ecole Normale Supérieure, PSL Research University [3]UC Berkeley
[*]Indicates equal contribution. Equal authors are listed in alphabetical order.

Abstract

Calibration—the problem of ensuring that predicted probabilities align with observed class frequencies—is a basic desideratum for reliable prediction with machine learning systems. Calibration error is traditionally assessed via a divergence function, using the expected divergence between predictions and empirical frequencies. Accurately estimating this quantity is challenging, especially in the multiclass setting. Here, we show how to extend a recent variational framework for estimating calibration errors beyond divergences induced induced by proper losses, to cover a broad class of calibration errors induced by L_p divergences. Our method can separate over- and under-confidence and, unlike non-variational approaches, avoids overestimation. We provide extensive experiments and integrate our code in the open-source package `probmetrics`[1] for evaluating calibration errors.

1 INTRODUCTION

We study the problem of calibration in classification. Let $\mathcal{X}$ be an arbitrary sample space and $\mathcal{Y}$ the space of one-hot-encoded categorical outcomes $\mathcal{Y} = \{y \in \{0,1\}^k \mid \sum_{i=1}^k y_i = 1\}$. Consider $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ such that $(X, Y) \sim \mathbb{P}$ for some unknown probability distribution $\mathbb{P}$. The goal of classification is to predict Y using information from the feature X.

In modern machine learning, we usually tackle this class of problems using methods that make continuous predictions in the probability simplex $\Delta_k = \{p \in [0,1]^k \mid \sum_{i=1}^k p_i = 1\}$. The model $f : \mathcal{X} \to \Delta_k$ not only predicts the most likely class but produces a probability vector assigning a score in $[0,1]$ to each class, evaluating the likelihood of observing the associated outcome. Such a classifier is called *calibrated* when

$$\mathbb{E}[Y|f(X)] = f(X) \quad \mathbb{P}\text{-almost surely.}$$

Since the outcome vector Y is one-hot encoded, the conditional expectation $C \coloneqq \mathbb{E}[Y|f(X)]$ is a probability vector in Δ_k with, $C_i = \mathbb{P}(Y = e_i|f(X))$. With this in mind, calibration indicates that the scores produced by the model $f(X)$ align with "real world" probabilities. Calibration is thus a desirable property, making model predictions interpretable for the end user.

Unfortunately, it is often observed that machine learning classifiers are not calibrated "out of the box" and tend to produce unreliable predictions (Zadrozny and Elkan, 2002; Guo et al., 2017; Berta et al., 2025b). This lack of calibration can be quantified via calibration error,

$$\mathrm{CE}_d(f) = \mathbb{E}[d(f(X), C)]\,, \tag{1}$$

where d can be any divergence function. In the binary setting $\mathcal{Y} = \{0,1\}$, $f : \mathcal{X} \to [0,1]$, the L_1 calibration error is often used:

$$\mathrm{CE}_{|\cdot|}(f) = \mathbb{E}[|f(X) - C|]\,. \tag{2}$$

Estimating calibration error is challenging as it requires approximating the conditional expectation C, with f continuous. In the binary case $f(X) \in [0,1]$, this is often done by binning the $[0,1]$ interval and computing the gap between predictions and average outcome for each bin, resulting in an estimator called expected calibration error (ECE, Naeini et al., 2015). ECE is biased and inconsistent (Kumar et al., 2019; Roelofs et al., 2022) and raises the problem of choosing the number of bins. An additional difficulty is that binning the simplex suffers from the curse of dimensionality when there are more than two classes. Given these difficulties, it is common to resort to computing

[1]`https://github.com/dholzmueller/probmetrics`

Workshop "Towards Trustworthy Predictions: Theory and Applications of Calibration for Modern AI" at AISTATS 2026, Tangier, Morocco.

the calibration error of the top class only, in a one-versus-rest fashion (Guo et al., 2017). Alternatively, Popordanoska et al. (2024) suggest approximating the calibration scores using kernel functions. Widmann et al. (2019) evaluate multiclass calibration error via an alternative variational formulation, approximating test functions using kernels.

Contributions. In this paper, we extend the calibration error estimator introduced by Berta et al. (2025a) to any binary or multiclass L_p calibration error. These estimators come with two main benefits:

- Using cross-validation to estimate the recalibration function and an evaluation based on proper losses guarantees that we lower bound, in expectation, the true calibration error.
- As we demonstrate empirically, our approach converges to the true calibration error faster than classical, binning-based, estimators.

2 ESTIMATING PROPER CALIBRATION ERRORS

Proper calibration errors. A family of calibration errors naturally appear in a well-known decomposition of the risk (expected loss) of the classifier $\mathbb{E}[\ell(f(X), Y)]$, when the loss function ℓ used is a proper loss (Gneiting and Raftery, 2007). Specifically, Bröcker (2009) showed that for any proper loss ℓ, the risk decomposes as

$$\mathbb{E}[\ell(f(X), Y)] = \mathbb{E}[d_\ell(f(X), C)] + \mathbb{E}[e_\ell(C)]\,, \qquad (3)$$

with $d_\ell(p, q) = \ell(p, q) - \ell(q, q)$ and $e_\ell(p) = \ell(p, p)$ the divergence and entropy functions induced by ℓ.

Looking at the first term in this decomposition, we recognize a calibration error (1), measured with the divergence function d_ℓ. Such calibration errors, induced by a proper loss ℓ, are called "proper calibration errors" (Gruber and Buettner, 2022).

A variational estimator. We describe a variational estimator for proper calibration error due to Berta et al. (2025a). The authors decompose the proper calibration error in (3) as follows:

$$\mathrm{CE}_{d_\ell}(f) = \mathbb{E}[\ell(f(X), Y)] - \min_{g \in \mathcal{H}} \mathbb{E}[\ell(g \circ f(X), Y)]\,, \quad (4)$$

where the min is taken over the set $\mathcal{H}$ of all measurable functions from Δ_k to Δ_k.

We denote by $g^\star$ the optimal recalibration function, uniquely defined by $g^\star(f(X)) = \mathbb{E}[Y|f(X)]$. For any strictly proper loss ℓ, $g^\star$ satisfies

$$g^\star \in \operatorname{argmin}_{g \in \mathcal{H}} \mathbb{E}[\ell(g \circ f(X), Y)]\,. \qquad (5)$$

Given an estimate $\hat{g}$ of $g^\star$, (4) reveals that a proper calibration error can be estimated by evaluating the risk of the initial model f minus the risk of $\hat{g} \circ f$,

$$\widehat{\mathrm{CE}}_{d_\ell}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i) - \ell(\hat{g} \circ f(X_i), Y_i)\,. \quad (6)$$

This provides a convenient way to estimate calibration error (1), in both binary and multiclass settings.

(5) shows that $g^\star$ can be estimated via empirical risk minimization of $\ell(g \circ f(X), Y)$ for any strictly proper loss ℓ. More generally, since $g^\star$ is the conditional expectation of the categorical variable Y given $f(X)$, an estimator $\hat{g}$ can be obtained with a classification algorithm targeting Y using $f(X)$ as a feature vector.

Obtaining a lower bound with cross-validation. This procedure requires estimating both the recalibration function $\hat{g}$ and the calibration error induced (6). We thus have two estimation tasks for a single set of samples. Re-using the data for both tasks exposes us to the classical pitfall of overfitting the recalibration function, meaning that the empirical risk of $\hat{g} \circ f$ on the data used to fit $\hat{g}$ might be far lower than the population risk of $g^\star \circ f$. In this case, the estimator (6) would over-estimate the true calibration error.

To circumvent this issue, Berta et al. (2025a) suggest learning $\hat{g}$ within a class of functions that is small enough to prevent overfitting, and/or using cross-validation. The benefit of using cross-validation is that learning $\hat{g}$ and estimating the calibration error with different samples guarantees that the calibration error is not over-estimated (in expectation). Indeed, for every function $\hat{g}_i$ learned on cross-validation split i,

$$\mathbb{E}[\ell(\hat{g}_i \circ f(X), Y)] \geq \mathbb{E}[\ell(g^\star \circ f(X), Y)]\,,$$

because $g^\star$ minimizes $\mathbb{E}[\ell(g \circ f(X), Y)]$. We then evaluate calibration error by estimating $\mathbb{E}[\ell(\hat{g} \circ f(X), Y)]$ on the hold-out split, and average the results. Up to the variance of the empirical expectation, this yields a lower bound on the true calibration error.

The better $\hat{g}$ approximates $g^\star$, the closer we get to the true calibration error. This motivates the use of well designed machine learning models to learn $\hat{g}$, which we investigate empirically in Section 4.

One apparent limitation of this procedure is that we are restricted to proper calibration errors like the squared Euclidean calibration error induced by the Brier score or the KL calibration error induced by the logloss. Any Bregman divergence can be recovered, but not distances induced by L_p norms, such as the usual L_1 binary calibration error (2). In the multiclass case, the L_1 calibration error is simply the

sum of the binary L_1 calibration error of each class against the rest, sometimes called "pairwise calibration error" $\mathbb{E}[\|f(X) - C\|_1] = \sum_{i=1}^{k} \mathbb{E}[|f(X)_i - C_i|]$. Another interesting multiclass extension is the L_2, or Euclidean calibration error $\mathbb{E}[\|f(X)-C\|_2]$, that is also not proper, and that cannot be estimated by binning the $[0, 1]$ interval. In the next section, we show how to use the estimator from Berta et al. (2025a) to estimate any L_p calibration error, extending an idea introduced by Braun et al. (2025).

3 ESTIMATING L_p CALIBRATION ERRORS

On the probability simplex, a proper loss is fully characterized by a concave function $H : \Delta_k \to \mathbb{R}$, via $\ell(u, v) = H(u) + \langle \delta H(u), v - u\rangle$, where $\delta H(u)$ denotes a super-gradient of H in u. The associated entropy function is $e_\ell(u) = H(u)$ and the divergence is $d_\ell(u, v) = H(u) - H(v) + \langle \delta H(u), v - u\rangle$. We refer the interested reader to Gneiting and Raftery (2007). Proper calibration errors CE_{d_ℓ} can be characterized this way but not L_p calibration errors.

Braun et al. (2025) show that by allowing the entropy function H (and thus the proper loss) to change for every $f(X)$, one can recover, in expectation, divergences that are not induced by a fixed proper loss. We show how to use this to evaluate a family of non-proper calibration errors: $\mathrm{CE}_{\|\cdot\|_p}(f)$, for any $p \geq 1$.

Proposition 1. *For $p \geq 1, z \in \Delta_k, Y \in \mathcal{Y}$, define*

$$\ell_{f(X)}(z, Y) \coloneqq \mathbb{1}_{z \neq f(X)} \langle \nabla_z \|z - f(X)\|_p, f(X) - Y\rangle \,,$$

where $\nabla_z \|z-f(X)\|_p = \mathrm{sign}(z-f(X)) \odot \frac{|z-f(X)|^{p-1}}{\|z-f(X)\|_p^{p-1}}$, with element-wise power and sign *in the numerator (for $p = 1$, $\nabla_z \|z - f(X)\|_p = \mathrm{sign}(z - f(X))$). Then,*

$$\mathrm{CE}_{\|\cdot\|_p}(f) = \mathbb{E}[\ell_{f(X)}(f(X), Y) - \ell_{f(X)}(g^\star \circ f(X), Y)] \,.$$

Proof. Let $H_{f(X)}(z) = -\|z - f(X)\|_p$, the associated proper loss is

$$\ell_{f(X)}(z, Y) = \langle \delta H_{f(X)}(z), Y - z\rangle + H_{f(X)}(z) \,.$$

There are multiple super-gradients of $H_{f(X)}(\cdot)$ in $f(X)$ but we set $\delta H_{f(X)}(f(X)) = 0$ so $\ell_{f(X)}(f(X), Y) = 0$.

Evaluating the variational calibration error (4) yields

$$\begin{aligned}
&\mathbb{E}[\ell_{f(X)}(f(X), Y)] - \mathbb{E}[\ell_{f(X)}(g^\star \circ f(X), Y)] \\
&\quad = -\mathbb{E}[\ell_{f(X)}(g^\star \circ f(X), Y)] \\
&\quad = -\mathbb{E}_{f(X)}[\mathbb{E}_{Y|f(X)}[\ell_{f(X)}(g^\star \circ f(X), Y)]] \\
&\quad = -\mathbb{E}_{f(X)}[\ell_{f(X)}(g^\star \circ f(X), C)] \\
&\quad = -\mathbb{E}[\ell_{f(X)}(C, C)] \\
&\quad = -\mathbb{E}[H_{f(X)}(C)] \\
&\quad = \mathbb{E}[\|C - f(X)\|_p] \,,
\end{aligned}$$

we recover the L_p calibration error. For a given $p \geq 1$, the proper loss induced is

$$\ell_{f(X)}(z, Y) = \langle \nabla_z \|z - f(X)\|_p, z - Y\rangle - \|z - f(X)\|_p$$

for every $z \neq f(X)$. The expression for $\nabla_z \|z-f(X)\|_p$ follows from a simple computation. Noticing that

$$\|z - f(X)\|_p = \langle \nabla_z \|z - f(X)\|_p, z - f(X)\rangle \,,$$

the loss simplifies to

$$\ell_{f(X)}(z, Y) = \langle \nabla_z \|z - f(X)\|_p, f(X) - Y\rangle \,,$$

which concludes the proof. □

In practice, to estimate the L_p calibration error of a classifier f, we fit $(\hat{g}_j)_{1 \leq j \leq k}$ with k-fold cross-validation. We then compute the calibration error on each holdout fold $\mathcal{I}_j$ with

$$\widehat{\mathrm{CE}}_{\|\cdot\|_p}(f)_j = -\frac{1}{|\mathcal{I}_j|} \sum_{i \in \mathcal{I}_j} \ell_{f(X_i)}(\hat{g} \circ f(X_i), Y_i) \,,$$

and average the k estimates obtained. We provide details on the procedure in Appendix A.

Remark 1. *This procedure applies not only to L_p norms but to any convex distance function by setting $d(f(X), z) = H_{f(X)}(z)$ such that $H_{f(X)}$ is minimized at 0 in $f(X)$, see Appendix B.*

Remark 2. *In Appendix C, we demonstrate how to use this procedure to evaluate over- and under-confidence separately for a more refined analysis of classifier's predictions. In the multiclass setting we use top class over- / under-confidence.*

Remark 3. *Another well known variational formulation co-exists for calibration error, see for example Kakade and Foster (2008). For $p = 2$, $\mathrm{CE}_{\|\cdot\|_2} = \sup_{h:\forall z \|h(z)\|_2 \leq 1} \mathbb{E}[\langle h(f(X)), Y - f(X)\rangle]$, it coincides with our formulation via $h(f(X)) = \nabla_z \|z - f(X)\||_{z=g(f(X))}$, but learning g instead of h allows the use of existing classifiers.*

4 EXPERIMENTS

We update the MetricsWithCalibration class in the probmetrics package that was realised by Berta et al. (2025a) to allow computing L_p calibration errors variationally, for any p, over- and under- confidence and top class errors.

Obtaining a lower bound with cross-validation. We illustrate the benefit of using cross-validation with a simple experiment on binary synthetic datasets[2]

[2]Our package is accessible at `https://github.com/dholzmueller/probmetrics` and the code for the experiments at `https://github.com/ElSacho/Evaluating_Lp_Calibration_Errors`

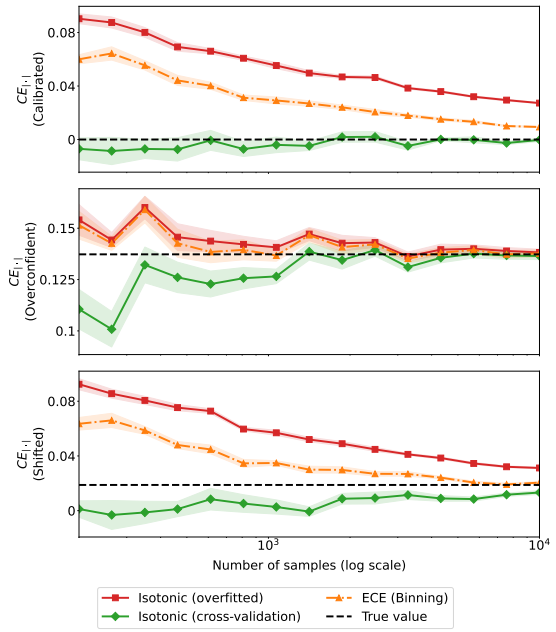

Figure 1: Estimated $\text{CE}_{|\cdot|}$ by number of samples when the predictions are calibrated (top), over-confident (middle), or shifted by a small parameter (bottom).

(multiclass results are in Appendix D). We compare applying our procedure to estimate $\text{CE}_{|\cdot|}$, using isotonic regression to learn $\hat{g}$, with and without cross-validation. As shown in Figure 1, using cross validation returns a lower bound on the true $\text{CE}_{|\cdot|}$. In contrast, without cross validation, isotonic regression over-fits and the calibration error estimate is pessimistic, particularly with few samples and when the underlying predictor is already well-calibrated. For completeness, we also report the calibration error estimate obtained with the standard ECE estimator described in the introduction, with 15 equal sized bins. It tends to over-estimates calibration error as well. We provide missing details on these experiments, along with additional results in Appendix D. While we use a different proper loss here, Dimitriadis et al. (2021) suggest estimating calibration error with isotonic regression without cross validation.

Approaching the true calibration error with better classifiers. While our procedure always yields a lower bound in expectation, how close our estimate is from the true calibration error is fundamentally linked to how well the underlying classifier learns to predict Y given $f(X)$. For this reason, one could be tempted to use highly tuned classifiers. However, since our end goal is to compute a metric, which should remain as fast as possible, we limit ourselves to classifiers that can be fitted in a few seconds. Following recent benchmarks (Holzmüller et al., 2024; Erickson et al., 2025), we select a suite of efficient models (we provide architectural details in Appendix E).

	Avg. % of max CE				
	Binary		Multiclass		
Classifier	$\text{CE}_{\lvert\cdot\rvert}$	$\text{CE}_{(\cdot)^2}$	$\text{CE}_{\lVert\cdot\rVert_2}$	$\text{CE}_{\lVert\cdot\rVert_2^2}$	Avg. time per 1K samples [s]
TabICLv2	$71.6_{1.7}$	$57.5_{2.3}$	$\mathbf{72.7}_{3.3}$	$\mathbf{66.7}_{3.8}$	$3.7_{0.1}$ (GPU)
RealTabPFN-2.5	$70.5_{1.7}$	$52.8_{2.4}$	$72.5_{3.1}$	$59.5_{3.9}$	$4.7_{0.1}$ (GPU)
WS CatBoost (+SMS)	$\mathbf{72.9}_{1.7}$	$\mathbf{59.4}_{2.2}$	$61.9_{3.3}$	$60.5_{3.9}$	$4.2_{0.1}$
WS LGBM (+SMS)	$70.3_{1.8}$	$58.1_{2.4}$	$59.5_{3.3}$	$54.2_{3.8}$	$3.3_{0.1}$
CatBoost (+SMS)	$70.3_{1.6}$	$36.1_{2.2}$	$71.7_{2.7}$	$40.7_{3.9}$	$8.9_{0.3}$
LightGBM (+SMS)	$65.7_{1.6}$	$29.2_{2.1}$	$68.3_{2.7}$	$33.8_{3.7}$	$4.3_{0.1}$
Nadaraya-Watson	$52.4_{2.1}$	$36.6_{2.4}$	$67.9_{2.9}$	$39.8_{4.1}$	$\mathbf{0.0}_{0.0}$
TS	$60.5_{2.2}$	$57.4_{2.6}$	$37.5_{3.5}$	$32.6_{4.2}$	$0.1_{0.0}$
Isotonic	$66.2_{1.7}$	$42.2_{2.4}$	$42.7_{2.9}$	$21.1_{2.9}$	$\mathbf{0.0}_{0.0}$
PartitionWise	$62.0_{1.6}$	$23.4_{2.0}$	$54.3_{2.7}$	$10.9_{2.4}$	$0.1_{0.0}$

Table 1: **CE recovered by different methods**, relative to the highest value among all methods for binary (left) and multiclass (right) experiments. CEs are averaged over 10 runs. The lowercase number is the standard error across all datasets. Methods are ranked based on the highest average over the four columns, best values per columns are in bold.

We evaluate the effectiveness of these classifiers in approximating different binary and multiclass calibration errors. To do so, we use the out-of-fold validation predictions stored in TabRepo (Salinas and Erickson, 2024) for 58 binary datasets with predictions from 6 different models and 25 multiclass datasets with predictions from 5 different models. For each experiment (dataset-model pair), we estimate calibration errors using 5-fold cross-validation. Given that our framework guarantees a lower bound on the true CE_d, higher estimated values indicate better estimation, giving a simple way to compare the different classifiers considered. Table 1 summarizes the average percentage of the largest calibration error estimate recovered by each classifier (larger is better), alongside the average time to compute each metric per 1,000 samples. We average values over ten independent runs and over all dataset-model pairs. To filter out well calibrated models, we remove experiments for which the largest estimated $\text{CE}_{|\cdot|}$ (binary) or $\text{CE}_{\lVert\cdot\rVert_2}$ (multiclass) is below 0.02.

Overall, the state-of-the-art classifiers TabICLv2 and RealTabPFN-2.5 recover the most calibration error. However, these models run on the GPU, which limits their usability as default options. We also experiment with widely used gradient boosting methods, namely CatBoost and LightGBM, comparing two training strategies: (i) standard training, which fits a new classifier from scratch using the raw predictions $f(X)$, and (ii) a strategy in which the model is warm-started using initial un-calibrated logits (denoted WS in the table). We fit these models using inner cross-validation for early stopping and post-hoc calibration with structured matrix scaling (multiclass) and quadratic scaling (binary) (Berta et al., 2025b). Both strategies yield comparable results for

non-proper calibration errors. Proper calibration errors however, require approximating more closely the true recalibration function, and logit initialization consistently improves performance.

Finally, we also evaluate commonly used estimators, including Nadaraya-Watson, temperature scaling, isotonic regression (one-versus-rest) and partition-wise estimators. While these methods provide faster estimates of the calibration error, this computational advantage comes at the cost of reduced accuracy, particularly for proper classification metrics.

In light of these findings, we recommend the logit-initialized CatBoost classifier as the default model in our package.

References

Banerjee, A., Guo, X., and Wang, H. (2005). On the optimality of conditional expectation as a bregman predictor. *IEEE Transactions on Information Theory*, 51(7):2664–2669.

Berta, E., Holzmüller, D., Jordan, M. I., and Bach, F. (2025a). Rethinking early stopping: Refine, then calibrate. *arXiv preprint arXiv:2501.19195*.

Berta, E., Holzmüller, D., Jordan, M. I., and Bach, F. (2025b). Structured matrix scaling for multi-class calibration. *arXiv preprint arXiv:2511.03685*.

Braun, S., Holzmüller, D., Jordan, M. I., and Bach, F. (2025). Conditional coverage diagnostics for conformal prediction. *arXiv preprint arXiv:2512.11779*.

Bröcker, J. (2009). Reliability, sufficiency, and the decomposition of proper scores. *Quarterly Journal of the Royal Meteorological Society*, 135(643):1512–1519.

Dimitriadis, T., Gneiting, T., and Jordan, A. I. (2021). Stable reliability diagrams for probabilistic classifiers. *Proceedings of the National Academy of Sciences*, 118(8):e2016191118.

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., and Smola, A. (2020). Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*.

Erickson, N., Purucker, L., Tschalzev, A., Holzmüller, D., Desai, P. M., Salinas, D., and Hutter, F. (2025). Tabarena: A living benchmark for machine learning on tabular data. In *International Conference on Machine Learning*.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Grinsztajn, L., Flöge, K., Key, O., Birkel, F., Jund, P., Roof, B., Jäger, B., Safaric, D., Alessi, S., Hayler, A., et al. (2025). TabPFN-2.5: Advancing the state of the art in tabular foundation models. *arXiv preprint arXiv:2511.08667*.

Gruber, S. and Buettner, F. (2022). Better uncertainty calibration via proper scores for classification and beyond. In *Advances in Neural Information Processing Systems*.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning*.

Holzmüller, D., Grinsztajn, L., and Steinwart, I. (2024). Better by default: Strong pre-tuned MLPs and boosted trees on tabular data. In *Advances in Neural Information Processing Systems*.

Kakade, S. M. and Foster, D. P. (2008). Deterministic calibration and nash equilibrium. *Journal of Computer and System Sciences*, 74(1):115–130.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*.

Kumar, A., Liang, P. S., and Ma, T. (2019). Verified uncertainty calibration. *Advances in Neural Information Processing Systems*.

Nadaraya, E. A. (1964). On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

Naeini, M. P., Cooper, G., and Hauskrecht, M. (2015). Obtaining well calibrated probabilities using Bayesian binning. In *AAAI Conference on Artificial Intelligence*.

Popordanoska, T., Gruber, S. G., Tiulpin, A., Buettner, F., and Blaschko, M. B. (2024). Consistent and asymptotically unbiased estimation of proper calibration errors. In *International Conference on Artificial Intelligence and Statistics*.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2018). Catboost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*.

Qu, J., Holzmüller, D., Varoquaux, G., and Morvan, M. L. (2026). TabICLv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*.

Roelofs, R., Cain, N., Shlens, J., and Mozer, M. C. (2022). Mitigating bias in calibration error estimation. In *International Conference on Artificial Intelligence and Statistics*.

Salinas, D. and Erickson, N. (2024). Tabrepo: A large scale repository of tabular model evaluations and its automl applications. In *AutoML Conference*.

Widmann, D., Lindsten, F., and Zachariah, D. (2019). Calibration tests in multi-class classification: A unifying framework. In *Advances in Neural Information Processing Systems*.

Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *International Conference on Knowledge Discovery and Data Mining*.

Supplementary Materials

A ALGORITHMIC PROCEDURE

We summarize the proposed algorithm as well as the cross-validation strategy in Algorithm 1.

Algorithm 1 Computing $\widehat{\mathrm{CE}_{d_\ell}}$.

Input: Data $\{(f(X_i), Y_i)\}_{i=1}^n$, number of folds $k \geq 2$, proper loss ℓ, classification method.
Partition the data: Randomly divide $\{1, \ldots, n\}$ into k approx. equal-sized folds $\{\mathcal{I}_1, \ldots, \mathcal{I}_k\}$.
for $j = 1$ to k **do**
  Define folds: $\mathcal{I}_{\text{val}}^{(j)} = \mathcal{I}_j, \quad \mathcal{I}_{\text{tr}}^{(j)} = \{1, \ldots, n\} \setminus \mathcal{I}_j$.
  Train classifier: Fit a classifier $g^{(j)}$ on $\{(f(X_i), Y_i) \mid i \in \mathcal{I}_{\text{tr}}^{(j)}\}$ using the specified method.
  Evaluate on validation fold: Using $i \in \mathcal{I}_{\text{val}}^{(j)}$, compute

$$\widehat{\mathrm{CE}_{d_\ell}}^{(j)} = \frac{1}{|\mathcal{I}_{\text{val}}^{(j)}|} \sum_{i \in \mathcal{I}_{\text{val}}^{(j)}} \ell(f(X_i), Y_i) - \ell(g^{(j)}(f(X_i)), Y_i).$$

end for
Aggregate across folds: $\widehat{\mathrm{CE}_{d_\ell}} = \sum_{j=1}^{k} \frac{|\mathcal{I}_j|}{n} \widehat{\mathrm{CE}_{d_\ell}}^{(j)}$.

B ESTIMATING GENERAL DISTANCES

The following proposition generalizes Theorem 1, showing that general distances $d(p, q)$ on the simplex can be written in a variational form whenever all functions $D_p(q) = d(p, q)$ are convex and minimized at p.

Proposition 2 (Multiclass extension of Proposition 3.1 in Braun et al. (2025))**.** *Let Δ_k be the probability simplex in $\mathbb{R}^k$. Let $f(X) \in \Delta_k$ be a baseline prediction. Let $D_{f(X)} : \Delta_K \to \mathbb{R}_{\geq 0}$ be a convex function minimized at $f(X)$, such that $D_{f(X)}(f(X)) = 0$.*

Let $\nabla D_{f(X)}$ be a subgradient of $D_{f(X)}$ satisfying $\nabla D_{f(X)}(f(X)) = 0$. Then, for $p \in \Delta_K$ and a one-hot encoded vector $Y \in \{e_1, \ldots, e_k\}$, the function

$$\ell_{f(X)}(p, Y) := -D_{f(X)}(p) - \langle Y - p, \nabla D_{f(X)}(p) \rangle ,$$

is a proper loss satisfying

$$\mathbb{E}_X[D_{f(X)}(C)] = \mathbb{E}[\ell_{f(X)}(f(X), Y)] - \inf_{g \in \mathcal{H}} \mathbb{E}[\ell_{f(X)}(g \circ f(X), Y)] ,$$

with $\mathcal{H}$ the set of all measurable functions from Δ_k to Δ_k and $C = \mathbb{E}[Y|f(X)]$.

Proof. First, $\ell_{f(X)}$ is a proper score because for all $p, q \in \Delta_K$, the definition of the expected score and the subgradient inequality for the convex function $D_{f(X)}$ yields:

$$\begin{aligned}
\mathbb{E}_{Y \sim q}[\ell_{f(X)}(p, Y)] &= -D_{f(X)}(p) - \langle \mathbb{E}_{Y \sim q}[Y] - p, \nabla D_{f(X)}(p) \rangle \\
&= -D_{f(X)}(p) - \langle q - p, \nabla D_{f(X)}(p) \rangle \\
&\geq -D_{f(X)}(q) \\
&= \mathbb{E}_{Y \sim q}[\ell_{f(X)}(q, Y)] .
\end{aligned}$$

Since we assumed $\nabla D_{f(X)}(f(X)) = 0$ and $D_{f(X)}(f(X)) = 0$, the score for predicting the baseline exactly is:

$$\ell_{f(X)}(f(X), Y) = -0 - \langle Y - f(X), 0 \rangle = 0 \ .$$

Hence, the expected excess risk of predicting p instead of the baseline $f(X)$, when the true distribution is p, is:

$$\begin{aligned}
\mathbb{E}_{Y\sim C}[\ell_{f(X)}(f(X), Y) - \ell_{f(X)}(C, Y)] &= \mathbb{E}_{Y\sim C}[-\ell_{f(X)}(C, Y)] \\
&= D_{f(X)}(C) + \langle \mathbb{E}_{Y\sim C}[Y] - C, \nabla D_{f(X)}(C) \rangle \\
&= D_{f(X)}(C) + \langle C - C, \nabla D_{f(X)}(p) \rangle \\
&= D_{f(X)}(C) \ .
\end{aligned}$$

Taking the expectation over X and using that

$$\mathbb{E}[\ell_{f(X)}(C, Y)] = \inf_{g \in \mathcal{H}} \mathbb{E}[\ell_{f(X)}(g \circ f(X), Y)] \ ,$$

(see for example Banerjee et al. (2005), or in the same setting of multiclass calibration Lemma A.6 in Berta et al. (2025a)), we obtain the desired result. □

C ESTIMATING OVER- AND UNDER-CONFIDENCE

Binary case. In the binary case, it is also possible to isolate over- and under-confidence. To do so, we adjust the proper loss to clip to $f(X)$ the values of the rectified prediction when they are over- or under-confident, following Braun et al. (2025). Specifically, given a proper loss ℓ, we define:

$$\begin{aligned}
\ell_{f(X),+}(p, y) &:= \ell(\mathbb{1}_{f(X)>1/2} \min\{p, f(X)\} + \mathbb{1}_{f(X)<1/2} \max\{p, f(X)\} + \mathbb{1}_{f(X)=1/2}/2, y) \\
\ell_{f(X),-}(p, y) &:= \ell(\mathbb{1}_{f(X)>1/2} \max\{p, f(X)\} + \mathbb{1}_{f(X)<1/2} \min\{p, f(X)\} + \mathbb{1}_{f(X)=1/2}/2, y)
\end{aligned}$$

that can respectively be used to estimates the over- and under-confidence of the classifier $f(\cdot)$.

Multiclass case. In the multiclass case, it is not clear how to define over- and under-confidence. One option is to adopt a one-versus-rest fashion and to evaluate the over- or under-confidence of the top class prediction using the losses that we just introduced.

Experiments. We illustrate the use of theses losses with simple synthetic experiments, with respectively over-confident predictions, under-confident predictions, or a mix of both. The mis-calibration functions we use are displayed in Figure 2, and the calibration errors obtained in Table 2. Using theses losses provides a refined analysis, detecting no under-confidence in the over-confident scenario, and revealing that the calibration error is fully determined by the over-confident component, that matches the true $\mathrm{CE}_{|\cdot|}$. Similar observations can be made in the under-confident experiment, and the effect of over- and under-confidence are well separated when both are present.

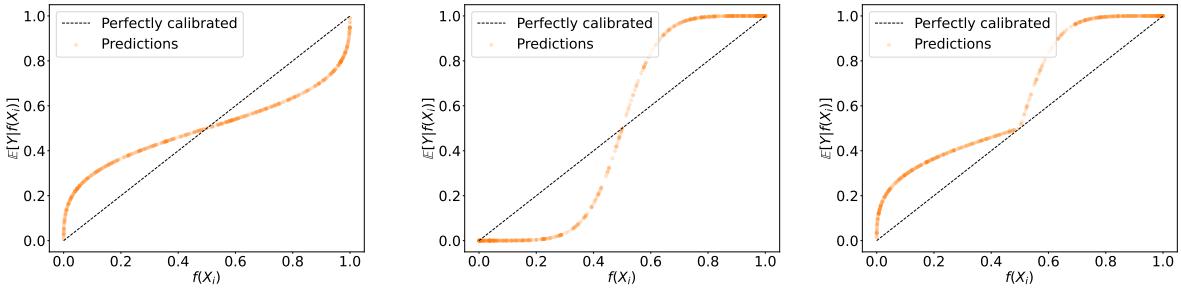

Figure 2: Different simulated mis-calibration scenarios. Predictions are either over-confident (left), under-confident (middle) or a mix of both (right).

	Over-confident	Under-confident	Both
$\widehat{\mathrm{CE}}_{\lvert Over(\cdot)\rvert}$	0.128	0.000	0.062
$\widehat{\mathrm{CE}}_{\lvert Under(\cdot)\rvert}$	0.000	0.126	0.063
$\widehat{\mathrm{CE}}_{\lvert\cdot\rvert}$	0.127	0.125	0.125
$\mathrm{CE}_{\lvert\cdot\rvert}$	0.129	0.128	0.128

Table 2: **CE recovered for the different over- and under-confident setups in Figure 2.** Values are averaged over 10 independent runs, negative values are then clipped to 0, and the calibration function is estimated with our Catboost+SMS model.

D SYNTHETIC EXPERIMENTS

We start by providing more details on the setup of our synthetic experiments. In both experiments, we first generate synthetic samples U on the simplex, to model the predictions $f(X)$. For binary classification, $U \sim \mathrm{Beta}(0.5, 0.5)$. For multiclass classification, $U \sim \mathrm{Dirichlet}(0.5 \times \mathbf{1}_d)$, where $\mathbf{1}_d = (1, 1, \ldots, 1)$ is a vector of d ones. This procedure tends to produce predictions near the boundaries of the simplex.

We then generate labels $Y \sim \mathcal{B}(g^\star(U))$, where $\mathcal{B}(p)$ is the Bernoulli distribution with parameter $p \in [0, 1]$, and $g^\star$ is the true calibration function, that we know in closed form in these synthetic experiments. In the multiclass case, this is generalized to $Y \sim \mathrm{Multinomial}(g^\star(U))$, where U is the initial prediction of the model and $g^\star(U)$ is the calibrated probability vector. The true calibration error is therefore $\mathrm{CE}_{\|\cdot\|_p} = \mathbb{E}[\|U - g^\star(U)\|_p]$. Since we have access to the calibration function $g^\star$, we can estimate this quantity, and we do so using $300,000$ samples. All experiments are repeated 10 times. The plots show the mean $\pm$ the standard error.

Binary. In the binary case, $u \in [0, 1]$, we test three different settings (results are presented in Section 4):

- The perfectly calibrated case: $g^\star(u) = u$.
- The over-confident case: $g^\star(u) = \mathrm{sigmoid}(0.4 \cdot \mathrm{logit}(u) + 0.3)$, where $\mathrm{logit}(u) = \log(u/(1-u))$.
- The shifted case: $g^\star(u) = \min(1, u + \varepsilon)$ with $\varepsilon = 0.02$.

Multivariate. In the multiclass case, $u \in \Delta_d$, and we experiment with $d = 3$ and $d = 10$ classes. We test three different settings:

- The perfectly calibrated case: $g^\star(u) = u$.
- The overconfident case: $g^\star(u) = \mathrm{sigmoid}(0.3 \cdot \log(u))$.
- The under-confident case: $g^\star(u) = \mathrm{sigmoid}(2 \cdot \log(u))$.

For the binning strategy, we group the predictions U into 30 bins by doing clustering on the samples U_i and compute the weighted average of the difference between confidence and accuracy across these bins. This is defined as $\sum_{i=1}^{30} \frac{\mathrm{Card}(B_i)}{n} \|\mathrm{acc}_i - \mathrm{conf}_i\|_2$, where B_i is the i-th bin, $\mathrm{acc}_i = \frac{1}{\mathrm{Card}(B_i)} \sum_{j \in B_i} Y_j$, and $\mathrm{conf}_i = \frac{1}{\mathrm{Card}(B_i)} \sum_{j \in B_i} f(X_j)$.

The results for $\mathrm{CE}_{\|\cdot\|_2}$ are shown in Figure 3. Again, our approach is the only one that consistently estimates the calibration error when the classifier is indeed calibrated. It always provides a lower bound, which converges faster to the true value in over-confident and under-confident settings.

E DETAILS ON CLASSIFIERS USED

Here, we provide more details on the classifiers used in Section 4. Most classifiers resemble the setup of Braun et al. (2025), and this section therefore closely follows their presentation.

Tabular foundation models. We evaluate RealTabPFN-2.5 (Grinsztajn et al., 2025) and TabICLv2 (Qu et al., 2026). These models can predict $\hat{g}(x_1^{\mathrm{test}}), \ldots, \hat{g}(x_n^{\mathrm{test}})$ with a single forward pass through a neural network that takes both the test input and the entire training set into account. They have been found to perform very well already without hyperparameter tuning (Erickson et al., 2025).

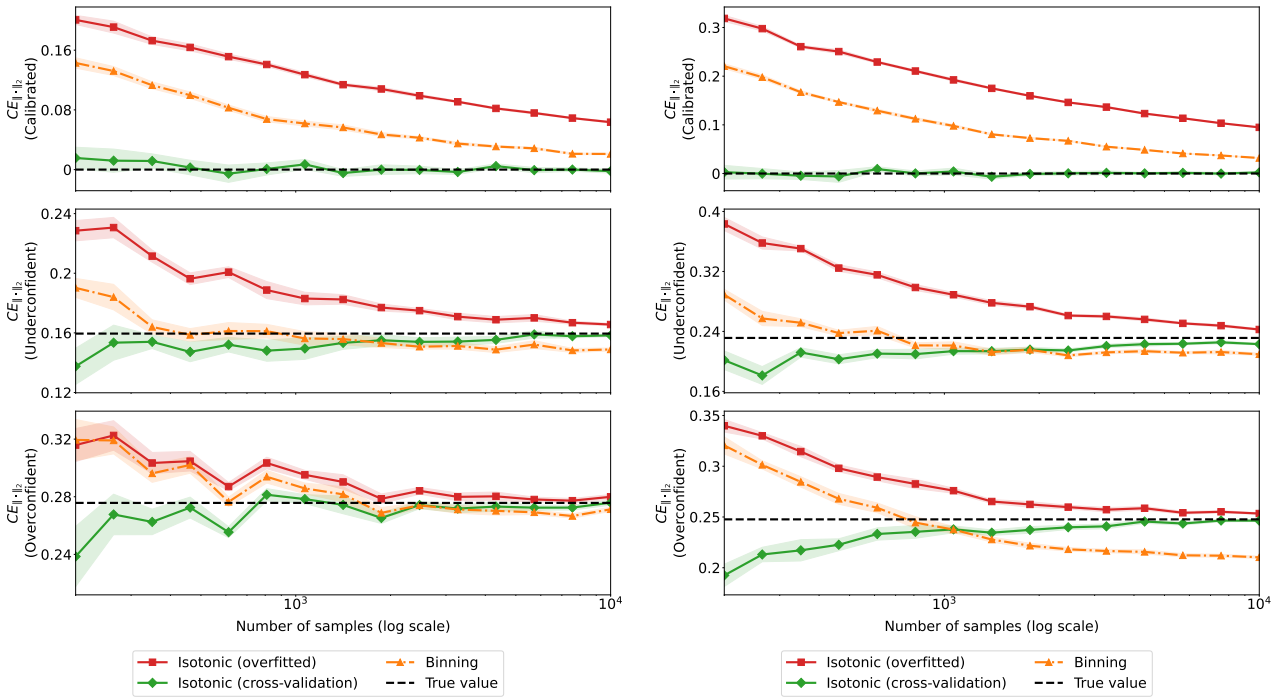

Figure 3: Estimation of $\mathrm{CE}_{\|\cdot\|_2}$ with binning, isotonic regression with over-fitting, and cross-validated isotonic regression on synthetic multiclass datasets with 3 classes (left) and 10 classes (right)

Gradient-boosted decision trees. We choose two representatives: CatBoost (Prokhorenkova et al., 2018) is known for its strong default performance. We use hyperparameters from AutoGluon (Erickson et al., 2020) and TabArena (Erickson et al., 2025), with 200 early stopping rounds instead of AutoGluon's custom early stopping logic. For a faster alternative, we use LightGBM (Ke et al., 2017) with cheaper hyperparameters, adapted from the tuned defaults of Holzmüller et al. (2024), reducing the number of early stopping rounds to 100 and using cross-entropy loss for early stopping (as for CatBoost). Both CatBoost and LightGBM are fitted in parallel with eight inner cross-validation folds for each outer cross-validation fold, following TabArena. The inner cross-validation folds are used for early stopping, and the final validation predictions are concatenated and used to fit a quadratic scaling (binary) or structured matrix scaling (multiclass) post-hoc calibration function (Berta et al., 2025b). Test set predictions are made based by applying the post-hoc calibration function to the average of the eight models' predictions.

Warm started gradient-boosted decision trees. We use the models described above, with additional warm-starting of the gradient-boosted decision trees using the non-calibrated logits. This reduces the complexity of the learning problem, so we reduce the tree count to 10. All other hyperparameters, including the inner cross-validation and quadratic scaling, remain consistent with previous experiments.

Nadaraya-Watson. Nadaraya-Watson (Nadaraya, 1964) estimation predicts the label by computing a locally weighted average of the training labels based on their distance in the feature space. The predictor relies on a radial basis function (RBF) kernel to determine these weights, and the kernel bandwidth parameter is fixed to 0.1. At inference time, the model evaluates the similarity between each new sample and all training instances, predicting a weighted mean where closer training points have a higher influence on the final output.

Temperature scaling. Temperature scaling (Guo et al., 2017) is a parametric calibration method that learns a single scalar parameter to scale the logits before the softmax or sigmoid function is applied. We use the implementation provided by Berta et al. (2025a). At inference time, the logits of each new sample are scaled by the learned parameter, which adjusts the confidence of the predicted probabilities without altering the original class predictions.

Isotonic regression. Isotonic regression is a non-parametric calibration technique that learns a piecewise constant function, monotonically increasing in $f(X)$ to map predictions to true empirical probabilities. We also use the implementation from Berta et al. (2025a). At inference time, the method takes the confidence score of a new sample and maps it through the learned step function to predict the calibrated probability. In the multiclass setting, we apply it in a one-versus-rest fashion.

PartitionWise. Partition-wise estimation first groups samples in the feature space and then predicts by using the average (one hot) label within each group. The predictor relies on KMeans to form the partitions, and the number of clusters is set to 15 for binary classification and 30 for multi-class. At inference time, each new sample is assigned to its nearest cluster and the model predicts the mean stored for that cluster. We use this classifier due to its similarity to binning.

Trade-offs. We chose to only fit boosted trees with inner cross-validation, both because they need validation sets for early stopping and because they are still reasonably fast with parallelization. The use of cross-validation also allows for the application of post-hoc calibration. Other methods might also benefit from post-hoc calibration, especially for non-L1 metrics, at the cost of higher runtime due to cross-validation.

Discussion and other options. We omit tabular neural networks trained from scratch from our comparison as they are relatively slow, especially on CPU, and their performance is suboptimal without tuning (Erickson et al., 2025). If runtime is less of a concern, for ideal sample-efficiency, automated machine learning methods such as AutoGluon (Erickson et al., 2020) can be employed, it combines multiple models and hyperparameter settings, ensembling, and post-hoc calibration. However, when these methods are used with time limits, the results may not be reproducible.

Hardware. We ran tabular foundation models on GPUs (NVIDIA V100) and the other models on CPUs (Cascade Lake Intel Xeon 5217 with 8 cores).