SmartCal: A Novel Automated Approach to Classifier Probability Calibration

Anonymous[1]

[1]Anonymous Institution

Abstract Accurate probability estimates are crucial in classification, yet widely used calibration methods like Platt and temperature scaling fail to generalize across diverse datasets. We introduce *SmartCal*, an AutoML framework that automatically selects the optimal post-hoc calibration strategy from a pool of 12 methods. Using a large-scale knowledge base of 172 datasets in multiple modalities and 13 classifiers, we show that no single calibrator is universally superior. *SmartCal* employs a meta-model trained on the meta-features of the calibration splits and classifier output to recommend the best calibration method for new tasks. Additionally, Bayesian optimization refines this selection process, outperforming standard baselines and random search. Experiments demonstrate that *SmartCal* systematically improves the calibration over existing approaches such as Beta Calibration and Temperature Scaling. This tool is freely available with a unified interface, simplifying the calibration process for researchers and practitioners.

1 Introduction

Accurate probability estimates are central to numerous real-world classification tasks, where reliable confidence scores inform decision-making in domains such as medical diagnosis, fraud detection, and autonomous vehicles (Vaicenavicius et al., 2019). A well-calibrated classifier should produce predicted probabilities that match the true likelihood of each class, enabling users to threshold scores for cost-sensitive classification, adapt to changing class priors, or simply trust the model's reported confidence (Cohen & Goldszmidt, 2004). In high-risk medical settings, an overconfident classifier may endanger patient safety by downplaying a rare but severe condition (Huang, Li, Macheret, Gabriel, & Ohno-Machado, 2020), while underconfident fraud detection systems can lead to unnecessary scrutiny of legitimate transactions (Leevy, Hancock, Khoshgoftaar, & Abdollah Zadeh, 2023).

Miscalibration often occurs when machine learning algorithms, especially complex models, systematically misjudge their own predictive probabilities (Pleiss, Raghavan, Wu, Kleinberg, & Weinberger, 2017). *Post-hoc* calibration techniques address this issue by converting raw classifier outputs into more reliable probabilities. Popular approaches include Platt scaling, isotonic regression, temperature scaling, beta calibration, and binning methods (Silva Filho et al., 2023). Each relies on a pre-trained classifier and fits a separate *calibration map* using dedicated validation data. However, these methods differ in assumptions, complexity, and dataset suitability (Huang et al., 2020; Vaicenavicius et al., 2019), posing a challenge for practitioners uncertain about which technique is best for their scenario (Widmann, Lindsten, & Zachariah, 2019).

Although prior work suggests that calibration effectiveness depends on factors like data distribution and model confidence, no single method consistently outperforms all others (Tao, Zhu, Guo, Dong, & Xu, 2023). In a large-scale study across 172 datasets (167 tabular, 3 image, 2 language) and 12 calibration techniques, we found that performance varied widely with respect to feature distribution, sample size, and class imbalance. Crucially, no single algorithm dominated across all settings, underscoring the context-dependent nature of calibration.

To address this variability, we propose *SmartCal*, an AutoML-based framework for classifier calibration. Central to *SmartCal* is a meta-model trained on the aforementioned large-scale knowledge base, which leverages dataset meta-features and uncalibrated classifier predictions to identify a short list of potentially optimal calibration algorithms. We further augment this recommendation process with a Bayesian optimization step that fine-tunes hyperparameters, more efficiently exploring the calibration space than naive alternatives like random selection. An open-source package provides a unified interface for the included algorithms, easing adoption by practitioners unfamiliar with calibration subtleties.

The contributions of this paper are summarized as follows:

- We introduce *SmartCal*, an AutoML framework for class probability calibration, based on a meta-model trained over a broad collection of datasets of different modalities and base classifiers.
- We provide a comprehensive, user-friendly package integrating twelve diverse post-hoc calibration algorithms under a single interface, accompanied by an open-source code release and reproducible results.[1]
- We present large-scale empirical evidence showing that each calibration method excels under specific conditions, motivating the need for an automated selection strategy.
- We demonstrate that *SmartCal* outperforms random search and baseline calibration methods in multiple experimental setups, validating its effectiveness in real-world scenarios.

2 Related Work

Post-processing calibration methods adjust probability outputs after a model is trained, aligning predicted probabilities with observed outcomes (Silva Filho et al., 2023). In contrast, some approaches incorporate calibration into the training objective (Kumar, Sarawagi, & Jain, 2018; Thulasidasan, Chennupati, Bilmes, Bhattacharya, & Michalak, 2019; Maher & Kull, 2021), but we do not explore these here.

Calibration Algorithms.. Non-parametric techniques like Empirical Binning (Naeini, Cooper, & Hauskrecht, 2015) partition predicted scores into bins and replace each bin's output with its empirical positive rate. Isotonic Calibration (Naeini & Cooper, 2016) similarly learns a non-decreasing function for mapping scores to probabilities. Both can be extended to multi-class tasks by applying them class-wise. Parametric methods, such as Platt Scaling (Platt et al., 1999) and Beta Calibration (Kull, Silva Filho, & Flach, 2017), fit transformations from logits to probabilities and can handle multi-class predictions on a per-class basis. Temperature Scaling (Guo, Pleiss, Sun, & Weinberger, 2017) is notably simple, using a scalar multiplier on logits, whereas Vector and Matrix Scaling (Guo et al., 2017) add per-class or matrix-level parameters. Dirichlet Calibration (Kull et al., 2019) generalizes these concepts for richer multi-class mappings, and further specialized methods include Multi-Class Uncertainty Calibration (Patel, Beluch, Yang, Pfeiffer, & Zhang, 2020), Mix-n-Match Calibration (Zhang, Kailkhura, & Han, 2020), Meta-Calibration (Ma & Blaschko, 2021), and Probability Calibration Trees (Leathart, Frank, Holmes, & Pfahringer, 2017). Despite this diversity, no single strategy dominates across all settings, with success often hinging on dataset-specific traits like class imbalance or label distribution (Mortier, Bengs, Hüllermeier, Luca, & Waegeman, 2023).

Automated Machine Learning (AutoML).. AutoML frameworks (e.g., Auto-sklearn (Feurer, Eggensperger, Falkner, Lindauer, & Hutter, 2022), TPOT (Olson & Moore, 2016), AutoGluon (Erickson et al., 2020), TabPFN (Hollmann et al., 2025), or LLM-driven pipelines (Hollmann, Müller, & Hutter, 2023)) have reduced the burden of model selection, hyperparameter tuning, and

[1] `https://anonymous.4open.science/r/SmartCal/README.md`

feature engineering (Singh & Joshi, 2022). However, they rarely incorporate calibration, leaving probability refinement as a manual step despite its importance for reliable decision-making (Cohen & Goldszmidt, 2004). This gap highlights the need for an AutoML-driven calibration approach. Rather than relying on fixed heuristics or user intuition, an automated system can adaptively select the best calibration method based on dataset properties and classifier behavior. Our work addresses this challenge by introducing *SmartCal*.

3 Problem Formulation

We consider a classification model M trained on a dataset D_{train} and a set of post-hoc calibration algorithms $\mathbf{C} = \{C^{(1)}, C^{(2)}, \ldots\}$, each with hyperparameters $\lambda \in \Lambda$. Let D_{cal} be a held-out calibration set used to learn or tune a calibration map, and let D_{test} be a disjoint test set on which we measure final performance. Our goal is to find a calibration algorithm $C_{\lambda}^{(i)}$ that minimizes a chosen calibration error metric (E) on D_{test}, subject to a constrained budget N for hyperparameter search. Formally, we seek

$$C_{\lambda}^{(i)*} \;=\; \arg \min_{C \in \mathbf{C},\, \lambda \in \Lambda} \mathrm{E}\big((C_{\lambda}, M), D_{\text{test}}\big), \tag{1}$$

where (C_{λ}, M) denotes the calibrated model obtained by applying calibrator C with configuration λ to the classifier M. The proposed *SmartCal* framework automates this process by learning a meta-model to recommend promising calibrators based on dataset meta-features and classifier outputs, then refining hyperparameters within the available budget N.

4 Methodology

In this section, we start by motivating the need for automated calibration tools by studying the diversity of best calibration methods (Section 4.1). Then, our proposed approach for automatically selecting and tuning post-hoc calibration algorithms is detailed. The workflow is divided into an *offline* phase (Section 4.2), where a large-scale knowledge base is constructed, and an *online* phase (Section 4.3), where the system exploits this knowledge base to provide calibration recommendations and hyperparameter tuning for new data.

4.1 Motivation for SmartCal: Diversity of Best Calibration Methods

In our large-scale study, we gathered the "best" post-hoc calibrator for each (dataset, classifier) pair according to a chosen calibration metric. Figure 1 illustrates the frequency with which each calibration algorithm emerged as the top performer. Although some methods appear more frequently than others, all except one algorithm achieve leading performance in at least a few cases. This pattern strongly suggests that no single calibration method uniformly outperforms the others across diverse data domains and base classifiers.

Relying on a single, standard calibrator may result in suboptimal performance across various problem instances. Hence, an automated selection approach is necessary to navigate the diverse range of algorithms and identify which one is most likely to excel in a particular scenario. Our proposed *SmartCal* framework addresses this gap by leveraging meta-features of the dataset and classifier outputs, supported by a knowledge base that accounts for the heterogeneous success patterns of distinct calibration methods.

4.2 Offline Phase: Building the Knowledge Base

Dataset Collection and Preparation. We gathered an extensive suite of public classification datasets from Kaggle [2], open UCI (Asuncion, Newman, et al., 2007) and OpenML (Vanschoren, Van Rijn,

[2] https://www.kaggle.com/

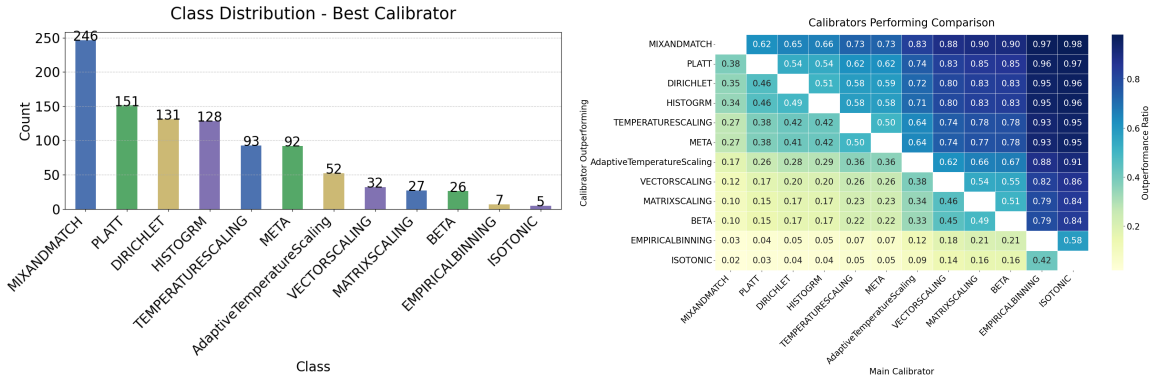

Figure 1: Distribution of top-ranked calibration algorithms demonstrating the absence of a universal "one-fits-all" approach. A histogram of the frequency of each calibrator having the best performance according to ECE in the knowledge base is shown on Left. A heatmap for the ratio of a calibration algorithm outperforms another one in the knowledge base is on the Right.

Bischl, & Torgo, 2014), covering tabular, language, and image domains. The datasets were selected, covering a wide range of sizes and count of classes. Each dataset was split into three partitions: a training set (for training the base classifier), a calibration set (for fitting post-hoc calibration algorithms), and a held-out test set (for final performance assessment) with percentages 60%, 20%, and 20%, respectively. Detailed information about these datasets, including their size, number of classes, and feature dimensions, is provided in Appendix A.2.

Classification Algorithms. On every dataset, we trained multiple classifiers according to domain relevance. All classification models are detailed in Appendix A.3. These classifiers produced raw (uncalibrated) probability estimates, which serve as the inputs for subsequent calibration.

Calibration Algorithms and Optimization Technique. We integrated twelve post-hoc calibration algorithms, described in Table 1 with the count of their hyper-parameters, and detailed in Appendix A.4, including widely used methods (Platt scaling, Temperature scaling, Isotonic regression, Beta calibration, various binning approaches) and more specialized algorithms for multi-class calibration (Vector/Matrix/Dirichlet scaling). Each algorithm had its own hyperparameter search space. Using the calibration set for each dataset-classifier pair, we performed a grid search to identify the hyperparameter configurations that optimize calibration performance (Section 4.2.1).

Table 1: Number of numerical and categorical hyperparameters for each calibration algorithm.

Calibration Algorithm	# Numerical	# Categorical
Empirical Binning	1	0
Isotonic Calibration	0	0
Beta Calibration	0	1
Temperature Scaling	3	0
Vector Scaling	2	0
Matrix Scaling	2	0
Dirichlet Calibration	2	0
Meta-Calibration	2	1
Platt Scaling	1	1
Histogram-based Calibrator	1	1
Adaptive Temperature Scaling	4	1
Mix-n-Match Calibration	0	2

4.2.1 Calibration Metrics and Knowledge Base Construction. After fitting each calibrator's grid-search configuration, we assessed calibration quality using five metrics summarized in Table 2.

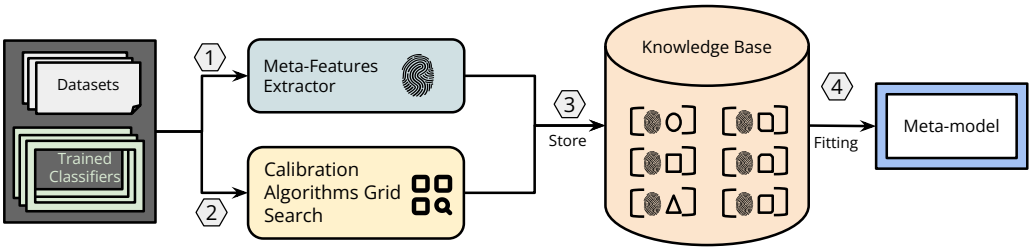

Figure 2: Meta-learning methodology: The meta-model is trained over a constructed knowledge base of the post-hoc calibrators performance over different classification algorithms.

These metrics capture different perspectives on how well predicted probabilities match observed frequencies (Arrieta-Ibarra, Gujral, Tannen, Tygert, & Xu, 2022).

Table 2: Calibration Metrics Used in the Evaluation.

Metric	Description
Max Calibration Error (MCE)	The maximum, over probability bins, of the absolute difference between the average predicted probability and the empirical frequency of the positive class.
Expected Calibration Error (ECE)	The average absolute difference between predicted probabilities and observed frequencies across all bins, providing a global measure of calibration.
Confidence Calibration Error (Conf ECE)	A variant of calibration error focusing on the predicted confidence (the maximum predicted probability), often relevant for single-label tasks.
Brier Score	A proper scoring rule that sums the squared difference between predicted probabilities and actual outcomes. Lower values indicate better calibration and accuracy combined.
Log Loss	Penalizes overconfident incorrect predictions more heavily. A widely used metric in probabilistic classification.

For each dataset-classifier pair, we identified the *best* calibration algorithm (and its hyperparameters) according to each of the metrics in Table 2, thus establishing multiple "winners" if using different metrics. We recorded these results in a *knowledge base* along with the **meta-features** of the calibration set, described next.

Meta-Feature Extraction. We extracted a set of meta-features from the calibration set, summarizing dataset properties (e.g., number of classes, class imbalance, distribution statistics) and classifier output behaviors (e.g., average predicted probability, entropy measures). Table 3 lists the meta-features used, along with the aggregation functions to convert raw measurements into stable scalars.

4.2.2 **Meta-Model Training**. We partitioned the knowledge base into a training set (80%) and a validation set (20%) in order to learn and evaluate our meta-model. Each entry in the knowledge base indicates which calibration algorithm performed best under specific conditions (dataset meta-features, classifier type, etc.). Instead of fitting a single multi-class model, we opted for a set of one-vs-all logistic regression classifiers, one for each potential "best" calibration method. During inference, each logistic regression model outputs a probability that its associated calibrator is the top performer for a given instance of meta-features. If one or more models produce probabilities exceeding a chosen threshold T, we include those corresponding calibrators in the recommended set. Lowering T encourages exploration by allowing more calibrators to be suggested, whereas raising T yields a more selective recommendation.

On the 20% validation split of the knowledge base, the final combined meta-model achieved an F1-score of 82.7% when $T = 0.4$. This threshold provided a practical balance between recommending

Table 3: Overview of the meta-features used in knowledge base construction.

Meta-feature	Details (Aggregations if any)
Calibration evaluation metric	
Classification model	
Classes count	
Num. of instances in calibration set	
Class imbalance ratio	
Classifier predictions entropy	
Classifier confidences	Mean, Median, Std, Var, Skewness, Kurtosis, Min, Max
Classification performance	Acc, Micro/Macro F1, Precision, Recall
Classifier calibration performance	ECE, MCE, Confidence ECE
Wasserstein distance KL Divergence Jensen Shannon Bhattacharyya	between predicted probabilities distributions and actual probability distributions in the calibration set. Aggregated with Mean, Median, Std, Var, Entropy, Skewness, Kurtosis, Min, Max

multiple potentially strong calibrators (exploration) and focusing on the few most likely to be optimal (exploitation).

4.3 Online Phase: Algorithm Recommendation and Tuning

In the online phase, our system operates on new datasets or tasks for which it aims to produce high-quality probability estimates via post-hoc calibration. The process illustrated in Figure 3 begins by extracting the same meta-features that were used during the offline phase. These meta-features describe key properties of the new calibration set, such as the shape and distribution of predicted probabilities, class imbalance, and dataset-specific statistics. Once the meta-features are collected, they are fed into the meta-model trained previously, which outputs a ranked list of recommended calibration algorithms. This recommendation is crucial for focusing subsequent efforts on the methods most likely to perform well in the current setting.

After receiving the ranked list of calibration algorithms, the system devotes a computational budget N iterations to tuning the top candidates. During this tuning stage, a focused hyperparameter optimization approach—commonly Bayesian optimization (Snoek, Larochelle, & Adams, 2012) is applied. Rather than exhaustively searching over all algorithms and parameter settings, the search targets the most promising methods and systematically explores their hyperparameter space. This targeted approach ensures an efficient balance between the exploitation of known good regions in the hyperparameter space and the exploration of less-tested configurations that may yield additional gains. Upon completing the optimization within the allocated time, the system selects the calibration algorithm and hyperparameter set, yielding the best results on a validation subset of the new calibration data. At this point, the chosen calibrator can be applied to the final test set or deployed in production. The resulting calibrated model benefits from the broader knowledge encapsulated in the offline-constructed database, combined with local fine-tuning informed by the current dataset's characteristics. Hence, the online phase substantially reduces the computational burden compared to a full grid search over all calibration algorithms space. At the same time, it leverages the historical experience encoded in the knowledge base, guiding the search for those approaches most likely to perform well.

5 Empirical Evaluation

This section presents our experimental setup, followed by a thorough evaluation of the proposed *SmartCal* framework from two angles. First, we compare the performance of the meta-model's algorithm recommendation against random algorithm selection. Second, we contrast the end-to-end

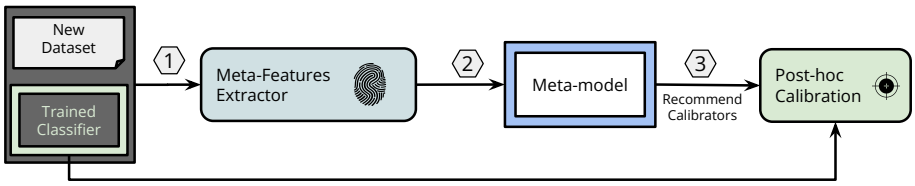

Figure 3: Online pipeline of the proposed methodology. The meta-model will be used to recommend post-hoc calibrators to be used with the trained classifier.

SmartCal pipeline (which includes meta-model recommendation plus hyperparameter tuning) with a brute-force random search over all calibration algorithms and parameter settings. We include a statistical analysis of the outcomes based on Wilcoxon signed-rank tests and provide a concluding discussion of key findings.

5.1 Experimental Setup

We evaluated *SmartCal* using 30 benchmark datasets spanning tabular, language, and image domains, all distinct from those used to build the knowledge base. Appendix A.2 provides full dataset details, while Appendix A.3 describes the domain-relevant classification models employed. Following the online phase outlined in Section 4.3, we extracted meta-features, obtained the meta-model's recommendations, and performed a constrained hyperparameter search on these recommended algorithms. All experiments were run on a machine with 4 vCPUs, 16 GB of memory, and Red Hat OS (Version 9.4) on an Intel Xeon(R) Gold 6138 CPU @ 2 GHz.

Calibration was evaluated on each dataset's *test* split using ECE, ensuring no overlap with the calibration sets. This protocol reflects true generalization and avoids overfitting to a single partition. We first compared the meta-model's ability to pick the best calibrator against a random selector, highlighting how effectively *SmartCal* identifies the most suitable algorithm for each dataset-classifier pair. Next, we benchmarked the entire *SmartCal* pipeline—meta-model plus hyperparameter tuning—against a random search baseline spanning all calibration algorithms and hyperparameters, as well as two commonly used methods: Temperature scaling and Beta calibration. This approach offers a broad view of *SmartCal*'s practical advantages and limitations.

5.2 Comparison of Meta-Model Recommendation vs. Random Algorithm Selection

We compared the meta-model's ability to select the lowest-ECE (best) calibrator on each (dataset, classifier) pair against a random selector. Figure 4 (left) summarizes three outcomes for their predictions: **Meta-Model Correct** (only the meta-model chooses the best calibrator), **Random Correct** (only random selection succeeds), and **Tie** (both select the same calibrator, correctly or incorrectly). The meta-model proves correct in the majority of cases, showing a marked advantage over random guessing for single-calibrator selection. We then examined whether the best calibrator lies within the meta-model's top-K recommendations. For each (dataset, classifier), the meta-model produced K suggestions, while the random selector provided K randomly chosen calibrators. Figure 4 (right plots the fraction of scenarios in which the true best calibrator is included, with K ranging from 1 to 7. The meta-model consistently achieves higher success rates than random selection at all K levels. A Wilcoxon signed-rank test on the paired results (meta-model vs. random selector) confirms that the meta-model's improvements are statistically significant ($p < 0.01$). This finding reflects the reliability of leveraging dataset meta-features and prior knowledge to guide calibration algorithm selection.

5.3 End-to-End SmartCal vs. Baselines

We compared our complete *SmartCal* pipeline (meta-model recommendation + hyperparameter tuning) against three baselines on 30 benchmarking datasets (each paired with multiple classifiers),

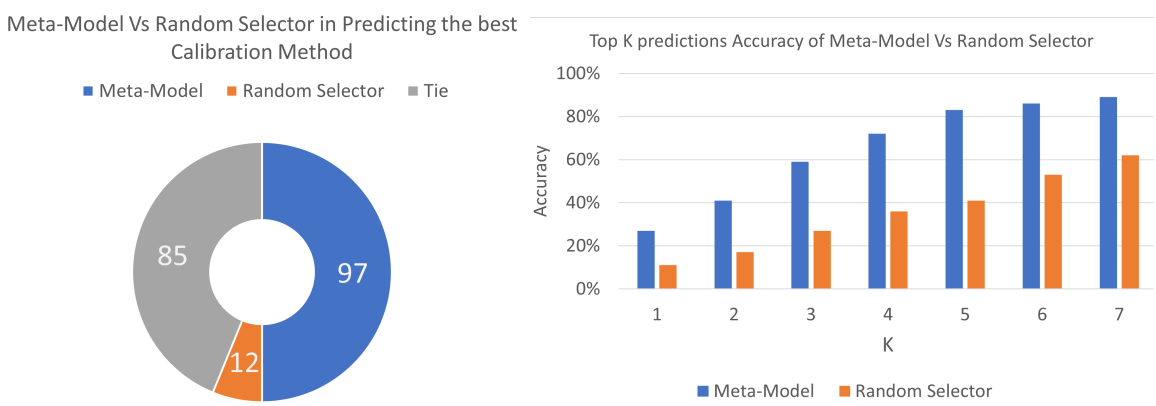

Figure 4: Comparison of meta-model vs. random selector. **Left**: Pie chart illustrating meta-model predictions correctness. **Right**: Bar chart of top-K inclusion from $K = 1$ to 7. The meta-model consistently outperforms random selection.

using Expected Calibration Error (ECE) as the key metric. The experiments were repeated thrice with different random seeds of dataset splits and average results with standard deviation are reported. The baselines include **(i)** Random Search across all calibration algorithms and their hyperparameter spaces (for $N = 10$ and $N = 30$ iterations), **(ii)** Temperature Scaling, and **(iii)** Beta calibration.

Table 4 summarizes the average ECE over all (dataset, classifier) combinations, along with standard deviations. The first two columns show the results for Random Search under two different iteration budgets ($N = 10$ and $N = 30$). The last two columns compare the Temperature scaling and Beta calibration methods, each used with its default hyperparameters or minimal tuning, to the proposed *SmartCal* approach.

Table 4: Average ECE and standard deviations across combinations of 30 benchmark datasets and 13 classifiers. Smaller is better.

	SmartCal(10)	**RS (10)**	**SmartCal(30)**	**RS (30)**	**Temp.**	**Beta**
Avg. ECE	0.0533 ± 0.112	0.0517 ± 0.115	0.0587 ± 0.084	0.0721 ± 0.070	0.1765 ± 0.107	
Avg. Rank	5.23 ± 1.44	6.31 ± 2.09	5.78 ± 1.87	6.84 ± 2.18	7.22 ± 2.46	

We observe that *SmartCal* yields the lowest mean ECE compared to all baselines, indicating more reliable calibration on average. Random Search with $N = 30$ performs closer to *SmartCal* than with $N = 10$, but still exhibits higher variability and slightly worse average ECE. Temperature scaling and Beta calibration, though simpler to deploy, show higher average miscalibration and cannot match the adaptiveness of *SmartCal*.

Overall, the proposed framework outperforms both fixed strategies (Temperature scaling and Beta calibration) and unstructured exploration (Random Search). The improvement is especially evident under small iteration budgets, where *SmartCal* makes more efficient use of limited trials. Complete per-dataset performance tables and additional plots are provided in Appendix X for reference, including detailed ECE values across each classifier–dataset combination.

5.4 Statistical Analysis

We ran a Friedman test followed by a Nemenyi post-hoc analysis (Demšar, 2006) on the average ranks of : *SmartCal* (10 vs. 30 iterations), Random Search (10 vs. 30 iterations), Temperature Scaling (TempScaling), and Beta Calibration (BetaCal). Figure 5 displays the critical difference (CD) diagram,

where lower rank values denote better calibration performance; methods whose intervals do not overlap the *CD* bar differ significantly at the chosen level.

As shown, both *SmartCal* variants achieve lower ranks than the baselines, especially with more iterations (SmartCal30). RandomSearch30 lies between *SmartCal* and the simpler fixed strategies, while RandomSearch10, TempScaling, and BetaCal occupy higher ranks. These findings reinforce our earlier observations that a meta-model–driven approach, combined with time-constrained hyperparameter tuning, yields more reliable calibration than either random or single-method baselines.

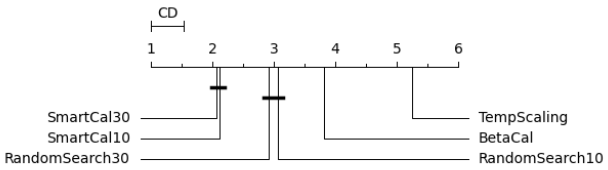

Figure 5: Critical difference diagram from the Friedman/Nemenyi analysis, comparing average ranks of *SmartCal*, Random Search, Temperature Scaling, and Beta Calibration. Lower ranks are better; overlapping intervals indicate no significant difference.

6 Conclusion, Limitations and Future Work

We presented *SmartCal*, an automated framework that learns when and how to apply different post-hoc calibration algorithms. By constructing a large-scale knowledge base through extensive experimentation and training a meta-model on dataset meta-features, *SmartCal* offers reliable algorithm recommendations and targeted hyperparameter tuning under a constrained budget. Empirical results showed that this data-driven approach substantially outperforms random search and other baselines, underscoring the value of informed calibrator selection.

Despite these positive outcomes, our work has a few limitations. First, *SmartCal* depends on an offline knowledge base that may not fully represent novel data distributions or emerging calibration algorithms. Second, our current study focuses mainly on standard supervised classification tasks, limiting the generality of our findings to other settings, such as structured outputs or multi-label tasks. Finally, some meta-features may not capture all intricacies of real-world datasets, which can lead to occasional mismatches between recommendation and actual performance.

Future work may extend *SmartCal* to specialized domains that require unique calibration metrics or meta-features. Enhancing meta-model interpretability could also clarify why certain calibrators are chosen. Additionally, integrating incremental updates to the knowledge base would help *SmartCal* adapt to new data and evolving calibration methods. Finally, exploring robustness to domain shifts and out-of-distribution samples could broaden the real-world applicability of the framework.

7 Broader Impact Statement

After careful reflection, the authors have determined that this work presents no notable negative impacts to society or the environment.

References

Arrieta-Ibarra, I., Gujral, P., Tannen, J., Tygert, M., & Xu, C. (2022). Metrics of calibration for probabilistic predictions. *Journal of Machine Learning Research*, *23*(351), 1–54.

Asuncion, A., Newman, D., et al. (2007). *Uci machine learning repository.* Irvine, CA, USA.

Cohen, I., & Goldszmidt, M. (2004). Properties and benefits of calibrated classifiers. In *European conference on principles of data mining and knowledge discovery* (pp. 125–136).

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, *7*(Jan), 1–30.

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

Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., & Hutter, F. (2022). Auto-sklearn 2.0: Hands-free automl via meta-learning. *Journal of Machine Learning Research*, *23*(261), 1–61.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International conference on machine learning* (pp. 1321–1330).

Hollmann, N., Müller, S., & Hutter, F. (2023). Llms for semi-automated data science: Introducing caafe for context-aware automated feature engineering. *CoRR*.

Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., … Hutter, F. (2025). Accurate predictions on small data with a tabular foundation model. *Nature*, *637*(8045), 319–326.

Huang, Y., Li, W., Macheret, F., Gabriel, R. A., & Ohno-Machado, L. (2020). A tutorial on calibration measurements and calibration models for clinical prediction models. *Journal of the American Medical Informatics Association*, *27*(4), 621–633.

Kull, M., Perello Nieto, M., Kängsepp, M., Silva Filho, T., Song, H., & Flach, P. (2019). Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. *Advances in neural information processing systems*, *32*.

Kull, M., Silva Filho, T., & Flach, P. (2017). Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial intelligence and statistics* (pp. 623–631).

Kumar, A., Sarawagi, S., & Jain, U. (2018). Trainable calibration measures for neural networks from kernel mean embeddings. In *International conference on machine learning* (pp. 2805–2814).

Leathart, T., Frank, E., Holmes, G., & Pfahringer, B. (2017). Probability calibration trees. In *Asian conference on machine learning* (pp. 145–160).

Leevy, J. L., Hancock, J., Khoshgoftaar, T. M., & Abdollah Zadeh, A. (2023). Investigating the effectiveness of one-class and binary classification for fraud detection. *Journal of Big Data*, *10*(1), 157.

Ma, X., & Blaschko, M. B. (2021). Meta-cal: Well-controlled post-hoc calibration by ranking. In *International conference on machine learning* (pp. 7235–7245).

Maher, M., & Kull, M. (2021). Instance-based label smoothing for better calibrated classification networks. In *2021 20th ieee international conference on machine learning and applications (icmla)* (pp. 746–753).

Mortier, T., Bengs, V., Hüllermeier, E., Luca, S., & Waegeman, W. (2023). On the calibration of probabilistic classifier sets. In *International conference on artificial intelligence and statistics* (pp. 8857–8870).

Naeini, M. P., Cooper, G., & Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 29).

Naeini, M. P., & Cooper, G. F. (2016). Binary classifier calibration using an ensemble of near isotonic regression models. In *2016 ieee 16th international conference on data mining (icdm)* (pp. 360–369).

Olson, R. S., & Moore, J. H. (2016). Tpot: A tree-based pipeline optimization tool for automating machine learning. In *Workshop on automatic machine learning* (pp. 66–74).

Patel, K., Beluch, W., Yang, B., Pfeiffer, M., & Zhang, D. (2020). Multi-class uncertainty calibration via mutual information maximization-based binning. *arXiv preprint arXiv:2006.13092*.

Platt, J., et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, *10*(3), 61–74.

Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J., & Weinberger, K. Q. (2017). On fairness and calibration. *Advances in neural information processing systems*, *30*.

Silva Filho, T., Song, H., Perello-Nieto, M., Santos-Rodriguez, R., Kull, M., & Flach, P. (2023). Classifier calibration: a survey on how to assess and improve predicted class probabilities. *Machine Learning*, *112*(9), 3211–3260.

Singh, V. K., & Joshi, K. (2022). Automated machine learning (automl): an overview of opportunities for application and research. *Journal of Information Technology Case and Application Research*, *24*(2), 75–85.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, *25*.

Tao, L., Zhu, Y., Guo, H., Dong, M., & Xu, C. (2023). A benchmark study on calibration. *arXiv preprint arXiv:2308.11838*.

Thulasidasan, S., Chennupati, G., Bilmes, J. A., Bhattacharya, T., & Michalak, S. (2019). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Advances in neural information processing systems*, *32*.

Vaicenavicius, J., Widmann, D., Andersson, C., Lindsten, F., Roll, J., & Schön, T. (2019). Evaluating model calibration in classification. In *The 22nd international conference on artificial intelligence and statistics* (pp. 3459–3467).

Vanschoren, J., Van Rijn, J. N., Bischl, B., & Torgo, L. (2014). Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, *15*(2), 49–60.

Widmann, D., Lindsten, F., & Zachariah, D. (2019). Calibration tests in multi-class classification: A unifying framework. *Advances in neural information processing systems*, *32*.

Zhang, J., Kailkhura, B., & Han, T. Y.-J. (2020). Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International conference on machine learning* (pp. 11117–11128).

Submission Checklist

1. For all authors...

   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]

   (b) Did you describe the limitations of your work? [Yes]

   (c) Did you discuss any potential negative societal impacts of your work? [N/A] As mentioned in Section 7

   (d) Did you read the ethics review guidelines and ensure that your paper conforms to them? `https://2022.automl.cc/ethics-accessibility/` [Yes]

2. If you ran experiments...

   (a) Did you use the same evaluation protocol for all methods being compared (e.g., same benchmarks, data (sub)sets, available resources)? [Yes]

   (b) Did you specify all the necessary details of your evaluation (e.g., data splits, pre-processing, search spaces, hyperparameter tuning)? [Yes]

   (c) Did you repeat your experiments (e.g., across multiple random seeds or splits) to account for the impact of randomness in your methods or data? [Yes]

   (d) Did you report the uncertainty of your results (e.g., the variance across random seeds or splits)? [Yes] In Table 4, we reported the average performance and standard deviation for each method.

   (e) Did you report the statistical significance of your results? [Yes] Both Sections 5.2 and 5.4 include statistical tests on the results significance.

   (f) Did you use tabular or surrogate benchmarks for in-depth evaluations? [Yes]

   (g) Did you compare performance over time and describe how you selected the maximum duration? [Yes] We conducted an experiment with increased number of iterations for both proposed method and baseline in Section 5.3.

   (h) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

   (i) Did you run ablation studies to assess the impact of different components of your approach? [Yes] Sections 5.2 includes the assessment of the meta-model separately from the end-to-end framework evaluated in Section 5.3

3. With respect to the code used to obtain your results...

   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results, including all requirements (e.g., `requirements.txt` with explicit versions), random seeds, an instructive `README` with installation, and execution commands (either in the supplemental material or as a URL)? [Yes] Included in the requirements and Readme files in the anonymized repository.

   (b) Did you include a minimal example to replicate results on a small subset of the experiments or on toy data? [Yes] Included in the Readme file examples for using our *SmartCal* package

   (c) Did you ensure sufficient code quality and documentation so that someone else can execute and understand your code? [Yes]

(d) Did you include the raw results of running your experiments with the given code, data, and instructions? [Yes] Raw results are included in our anonymized repository under raw folder.

(e) Did you include the code, additional data, and instructions needed to generate the figures and tables in your paper based on the raw results? [Yes] The code, knowledge base data and scripts used to generate the results can be found in our anonymized Repository.

4. If you used existing assets (e.g., code, data, models)…

(a) Did you cite the creators of used assets? [Yes]

(b) Did you discuss whether and how consent was obtained from people whose data you're using/curating if the license requires it? [N/A]

(c) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We are using public and open-source datasets.

5. If you created/released new assets (e.g., code, data, models)…

(a) Did you mention the license of the new assets (e.g., as part of your code submission)? [N/A] We have not published the code yet due to a double-blide review

(b) Did you include the new assets either in the supplemental material or as a URL (to, e.g., GitHub or Hugging Face)? [Yes] `https://anonymous.4open.science/r/SmartCal`

6. If you used crowdsourcing or conducted research with human subjects…

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

7. If you included theoretical results…

(a) Did you state the full set of assumptions of all theoretical results? [N/A]

(b) Did you include complete proofs of all theoretical results? [N/A]