Machine Learning
https://doi.org/10.1007/s10994-023-06336-7

Classifier calibration: a survey on how to assess and improve predicted class probabilities

Telmo Silva Filho[1,2] · **Hao Song**[2] · **Miquel Perello-Nieto**[2] · **Raul Santos-Rodriguez**[2] · **Meelis Kull**[3] · **Peter Flach**[2]

Received: 5 January 2022 / Revised: 26 January 2023 / Accepted: 7 April 2023

Abstract
This paper provides both an introduction to and a detailed overview of the principles and practice of classifier calibration. A well-calibrated classifier correctly quantifies the level of uncertainty or confidence associated with its instance-wise predictions. This is essential for critical applications, optimal decision making, cost-sensitive classification, and for some types of context change. Calibration research has a rich history which predates the birth of machine learning as an academic field by decades. However, a recent increase in the interest on calibration has led to new methods and the extension from binary to the multiclass setting. The space of options and issues to consider is large, and navigating it requires the right set of concepts and tools. We provide both introductory material and up-to-date technical details of the main concepts and methods, including proper scoring rules and other evaluation metrics, visualisation approaches, a comprehensive account of post-hoc calibration methods for binary and multiclass classification, and several advanced topics.

Keywords Classification · Calibration · Confidence · Uncertainty · Multiclass · Evaluation

1 Introduction and motivation

A K-class probabilistic classifier is *well-calibrated* if among test instances receiving a predicted K-dimensional probability vector $\mathbf{s}$, the class distribution is (approximately) distributed as $\mathbf{s}$. This property is of fundamental importance when using a classifier for cost-sensitive classification, for human decision making, or within an autonomous system. It means that the classifier correctly quantifies the level of uncertainty or confidence associated with its predictions. In a binary setting, scores given by a sufficiently calibrated classifier can be simply thresholded to minimise expected misclassification cost. Thresholds can also be derived to optimally adapt to a change in class prior, or to a combination of both. In

Editor: Eyke Hüllermeier.

Telmo Silva Filho, Hao Song and Miquel Perello-Nieto have contributed equally to this work.

Extended author information available on the last page of the article

contrast, for a poorly calibrated classifier the optimal thresholds cannot be obtained without optimisation.

Many machine learning algorithms are known to produce over-confident models, unless dedicated procedures are applied during training. The goal of *(post-hoc) calibration methods* is to use hold-out validation data to learn a *calibration map* for a previously trained model that transforms the model's predictions to be better calibrated. Many calibration methods for binary classifiers have been introduced, including logistic calibration (also known as 'Platt scaling'), various binning methods including isotonic calibration (also known as the ROC convex hull method), as well as more recent methods including beta calibration and Bayesian methods.

When we have more than two classes, calibration is generally more involved, as is often the case with multiclass classification. Multiclass calibration has mostly been approached by decomposing the problem into K one-vs-rest binary calibration tasks, one for each class. The predictions of these K calibration models form unnormalised probability vectors, which, after normalisation, may not be calibrated in a multiclass sense. Native multiclass calibration methods were introduced recently focusing on neural networks. These methods constitute various multiclass extensions of Platt scaling, adding a calibration layer between the logits of the neural network and the softmax layer.

The literature on post-hoc classifier calibration in machine learning is now sufficiently rich that it is no longer straightforward to obtain or maintain a good overview of the area, which was the main motivation for writing this survey. It grew out of a tutorial we presented at the 2020 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (see https://classifier-calibration.github.io). Our aim then and now is to provide both introductory material and up-to-date technical details of the main concepts and methods. Our focus is on the classical setup where the classifier is deployed in the same setting where it was calibrated, except for potential changes in class prior or misclassification costs. We do not cover methods to achieve robustness under conditions of other distributional shifts or out-of-distribution inputs (Ovadia et al., 2019). We try to do justice to historical developments and pay attention to important topics that are not as widely known as they deserve to be, such as proper scoring rules.

We also try to identify and clarify possible sources for confusion. The multiclass setting in particular introduces numerous subtleties that have not always been recognised or correctly dealt with in previous work. For example, some authors use the weaker notion of *confidence calibration*, which requires only that the classifier's predicted probability for what it considers the most likely class is calibrated. While this is perfectly valid in its own right, it isn't always appreciated by subsequent authors that this is a much weaker notion of calibration than the one defined informally in the first sentence of this survey. One can also observe variations in the evaluation metric used and in the way calibrated probabilities are visualised. Our main aim in this survey is hence to provide a unified perspective on the different methods and metrics for binary and multiclass calibration, giving each variation its proper place.

For the purpose of this survey we have developed a Python library which includes most of the functionalities presented in the following sections. PyCalib [1] implements several calibration metrics (eg. confidence and classwise ECE, and their MCE counterparts),

[1] https://classifier-calibration.github.io/PyCalib/.

common calibration methods (eg. Isotonic Calibration, Platt's Scaling, Binning calibration), a method to combine arbitrary classifiers and calibrators into a unified training and deployment pipeline, a one-vs-the-rest implementation to adapt any binary calibration method to the multiclass setting, and multiple visualisation tools to inspect visually the calibration quality as well as tools to get better insights about the learned calibration maps. As an illustration of the library's capabilities, most of the figures and results reported in this paper have been generated using PyCalib.

The survey is structured as follows. The next section will cover important notions and intuitions related to calibration. It aims to be at a more introductory level, leaving technical detail for later sections. Section 3 discusses proper scoring rules, an important general framework for evaluation of probability estimates. In Sect. 4 we present systematic overview of visualisation approaches and evaluation metrics for classifier calibration. Section 5 is devoted to a comprehensive account of post-hoc calibration methods for both the binary and multiclass scenarios. Section 6 covers hypothesis tests for calibration, and Sect. 7 concludes with a summary and outlook.

2 A brief introduction to classifier calibration

The origins of classifier calibration can be traced back to weather forecasting and meteorology. Here is what the UK Met Office website used to say about what their forecast probabilities mean:

> "[S]uppose the Met Office says that the probability of rain tomorrow in your region is 80%. They aren't saying that it will rain in 80% of the land area of your region, and not rain in the other 20%. Nor are they saying it will rain for 80% of the time. What they are saying is there is an 80% chance of rain occurring at any one place in the region, such as in your garden. [...] [A] forecast of 80% chance of rain in your region should broadly mean that, on about 80% of days when the weather conditions are like tomorrow's, you will experience rain where you are. [...] If it doesn't rain in your garden tomorrow, then the 80% forecast wasn't wrong, because it didn't say rain was certain. But if you look at a long run of days, on which the Met Office said the probability of rain was 80%, you'd expect it to have rained on about 80% of them. " (https://web.archive.org/web/20210928235732/https://www.metoffice.gov.uk/about-us/what/accuracy-and-trust/probability)

Note the phrase "a long run of days": determining the degree to which a forecaster is well-calibrated cannot be done on a per-forecast basis, but rather requires looking at a sufficiently large and diverse set of forecasts.

Table 1, reproduced from Hallenbeck (1920), groups 123 weather forecasts in 10 equal-width bins, and compares each bin with the actual empirical percentage of rain events. Figure 1 shows two possibilities of displaying this information graphically, comparing the predicted probabilities on the x-axis to empirical probabilities on the y-axis. A variety of such graphical representations can be found in the literature; they are generally known as *reliability diagrams*, 'reliability' being one of many terms used to denote goodness of calibration (Murphy & Winkler, 1977). We can see that most of the forecasts are slight under-estimates: for example, of the 15 forecasts in the 40–49% bin, $8/15 = 53\%$ were actual rain events. In the left graph this is shown by the vertical red line, which indicates how much the top of the blue bar extends above the diagonal

Table 1 Reproduced from Hallenbeck (1920)

Forecasted probability	Number of forecasts	Number of rains	Actual percentage
Precent			
Above 90	1	1	100
80 to 89	1	1	100
70 to 79	7	6	86
60 to 69	15	11	75
50 to 59	13	8	62
40 to 49	15	8	53
30 to 39	18	6	33
20 to 29	31	7	23
10 to 19	22	4	18
0 to 9	0	–	–

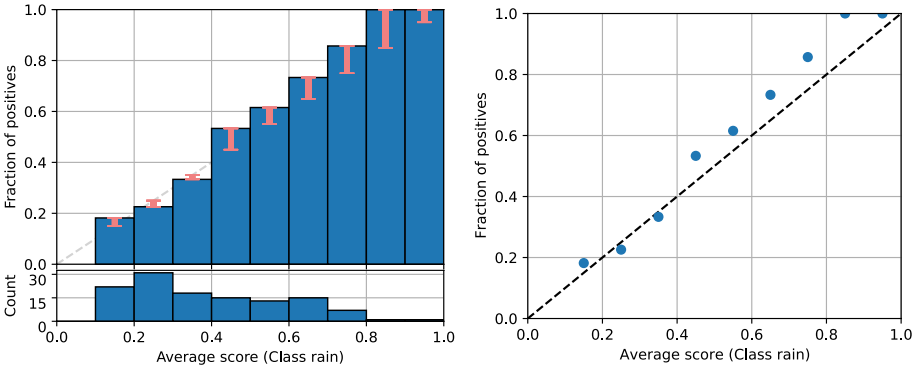


Fig. 1 Two different graphical representations of the forecasts in Table 1. **(Left)** The bottom graph gives a histogram of the probability forecasts. The top graph compares the forecast probabilities with the proportion of actual rain events in that bin. The red bars indicate to what extent these proportions are higher or lower than predicted. **(Right)** A simplified visualisation, only showing the proportion of rain events in a bin (y-axis) against the average prediction in the bin (x-axis)

(meaning that the actual proportion of rain events was higher than estimated). On the right graph this can be seen directly from how much the respective point is higher than the diagonal. Similarly, we see that two of the bins (20–29% and 30–39%) are very slight over-estimates: on the left graph the top of the blue bar is below the diagonal, which means that the actual proportion of rain events was lower than estimated.

The choice of ten equal-width bins is somewhat arbitrary, and it is informative to consider different choices. Merging pairs of adjacent bins results in five equal-width bins, as depicted in Fig. 2 (left). We see that all but the right-most bin averages are now slightly off-centre (we assumed that within each original bin all forecast probabilities are the same and equal to the bin centre: e.g., 25% in the 20–29% bin). Again, the pattern is that the higher forecast probabilities are about 10% too low. Figure 2 (right) shows what we get with only two bins. We have lost a bit too much information here, so bins shouldn't be taken too wide. On the other hand they shouldn't be taken too narrow

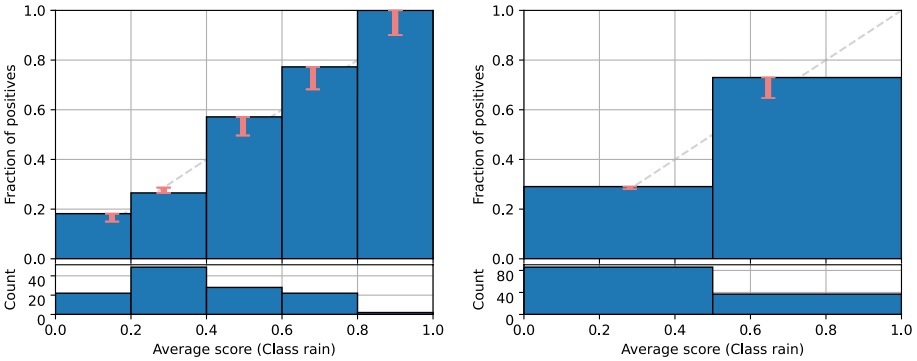


Fig. 2 Dividing the forecasts in Table 1 in five (left figure) and two (right figure) equal-width bins

either, in order to be able to measure the true proportion of rain events with sufficient resolution.

Taking the forecast event as a binary variable means that the calibration concept is directly applicable to binary classification. Here is an example from the typical machine learning task of learning a classifier for distinguishing 'junk' or 'spam' emails from regular ones:

> A prediction '70% chance of spam' should be spam 70% of the time.

And a generalisation to categorical variables with more than two values is equally straightforward, as this example from Fisher's famous Iris dataset shows:

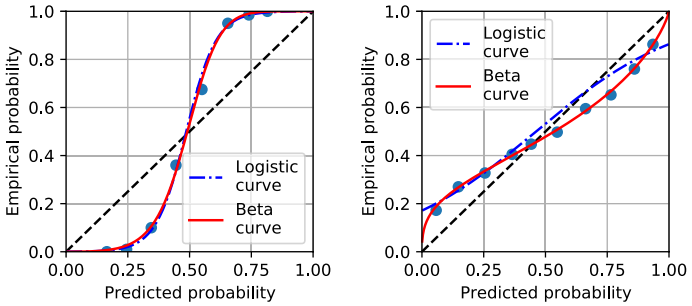
> A prediction '70% chance of setosa, 10% chance of versicolor and 20% chance of virginica' should be setosa/versicolor/virginica 70/10/20% of the time.

So, to sum up: *A predicted probability (vector) should match empirical (observed) probabilities.* In the language of predictive machine learning, given an instance space $\mathbb{X}$, a binary target space $\mathbb{Y} = \{+, -\}$, and a binary probabilistic classifier $\mathsf{f} : \mathbb{X} \to [0, 1]$, the binary classifier is *calibrated* if $\forall s \in [0, 1]$:

$$\mathsf{P}(Y = + \mid \mathsf{f}(\mathbf{X}) = s) = s$$

2.1 Why calibration matters

What are the benefits of using well-defined calibrated scales in general, and well-calibrated probabilities in particular? One obvious benefit of calibrated scales is that we can easily combine measurements that are expressed on the same scale without comparing apples and oranges. Another is that we can use standardised decision rules, e.g., defining a fever as a body temperature exceeding 100 degrees Fahrenheit. These benefits directly carry over to the class probability scenario, justifying decision rules such as predicting the class whose estimated probability exceeds 0.5 in binary classification, or the class with the highest predicted probability for multiclass classification. Importantly, there is a further benefit to using calibrated probabilities as it means we can adjust these decision rules in a straightforward way to account for different class priors or misclassification costs, as we will briefly discuss presently.



(a) Underconfidence (b) Overconfidence

Fig. 3 Examples of under- and overconfident classifiers. The axes are the same as for the previous figures: x-axis shows predicted probabilities and y-axis shows empirical probabilities. The dots represent the reliability diagram, and the lines show the best-fit logistic curve and beta curve. Note that the logistic sigmoid is a good fit for underconfident scores but not for overconfidence. The figure has been adapted from Niculescu-Mizil and Caruana (2005), and the beta curves have been added

Let $c = \frac{C(+|-)}{C(+|-)+C(-|+)}$ be the cost of a false positive in proportion to the combined cost of one false positive and one false negative. With these cost parameters the Bayes-optimal decision rule is to set the decision threshold to c. For instance, if false positives are 4 times as costly as false negatives then we set the decision threshold to $4/(4+1) = 0.8$ in order to only make positive predictions if we are sufficiently certain. Similar reasoning applies to changes in class priors:

- If we trained on balanced classes but want to deploy with 4 times as many positives compared to negatives, we lower the decision threshold to 0.2;
- More generally, if we trained for class ratio r and deploy for class ratio r' we set the decision threshold to $r/(r + r')$.

Cost and class prior changes can be combined in the obvious way; see Flach (2014) for details.

In summary, an important reason to be interested in calibrating our classifiers is that default decision rules such as predicting the class with the highest predicted probability are fully justified from a decision-theoretic point of view, and can easily be adapted to changes in class and cost distributions. In contrast, poorly calibrated classifiers such as naive Bayes will often perform sub-optimally with default decision rules. It is possible to learn a better decision rule for a given cost and class skew using, e.g., ROC analysis (Lachiche & Flach, 2003) but this would have to be repeated each time the skew changes. The great advantage of post-calibration is that it only needs to be done once – in a sense, it optimises all possible decision rules in one go.

2.2 Common forms of miscalibration, and how to fix them

There are many reasons why a probabilistic classifier might produce miscalibrated scores, and hence many ways in which miscalibration manifests itself, but two main types stand

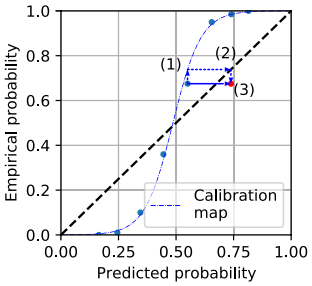


Fig. 4 Graphical illustration of how a well-fitted calibration map leads to near-perfect calibration. From left to right, the dotted arrows show (1) using a point's uncalibrated score s on the x-axis as input to the calibration map, (2) mapping the resulting output $\mathsf{c}(s)$ back to the diagonal, and (3) combine with the empirical probability of the point we started from. The closer the original point is to the fitted calibration map, the closer the calibrated point (in red) will be to the diagonal

out: underconfidence and overconfidence. In this section we look at some examples and introduce two mitigation techniques.

Figure 3, taken from Niculescu-Mizil and Caruana (2005), shows two typical examples. In both cases the points in the reliability diagram are far from the diagonal, although the effect is smaller in the figure on the right.

What do we see in the left figure? It is quite natural to give precedence to vertical information in such graphs, so you might be inclined to say "The points on the left are too low, and the points on the right are too high". But keeping in mind that the predicted probabilities on the x-axis are under our control but not the empirical probabilities on the y-axis, a more 'actionable' way to express this is

> The lower points are too far to the right, and the higher points are too far to the left. In other words, the points have a tendency to sit too close to the midpoint on the x-axis.

This is the typical pattern displayed by an *underconfident* classifier, which thinks it's worse at separating classes than it actually is.

Hence, to mitigate this, we need to pull predicted probabilities away from the centre. As the left plot in Fig. 3 shows, this can be neatly modelled by a sigmoidal logistic curve, which can be defined in parametric form as

$$\mathsf{c}(s;w,m) = \frac{1}{1+\exp(-w(s-m))}$$

Here, s is the uncalibrated score produced by the classifier and c gives the logistically calibrated score. m and w are the two parameters of the logistic family, which determine the point where $\mathsf{c}(s) = 0.5$ and the slope at that point, respectively. They can be estimated from the points in the reliability diagram in order to find the best fit. In this case they can be obtained analytically as the mean of the classwise average scores, and the difference between those two means in proportion to the score variance (Flach, 2012).

The fitted logistic curve thus establishes a *calibration map* which transforms uncalibrated scores s into calibrated scores $\mathsf{c}(s)$. Even though the calibration map doesn't map to empirical probabilities, plotting it over the reliability diagram allows us to see clearly that if we project each point in the reliability diagram vertically onto the calibration map, and replace its x-value with the corresponding y-value of the projection, we end up

with points on or very close to the diagonal, and hence near-perfectly calibrated scores (Fig. 4).

The logistic curve is widely used as a 'squashing function', compressing values on a possibly unbounded scale into the [0, 1] interval. Following Platt (2000) it is often used as a way to convert scores from a Support Vector Machine into probabilities. Technically this isn't calibration as SVM scores are not probabilities but (signed) geometric distances from the decision boundary – this is also called *scaling* to emphasise the difference (hence the popular name 'Platt scaling'). The parametric form of the logistic map can be derived from first principles by assuming that within each class the uncalibrated scores are normally distributed with the same variance.

The other common form of miscalibration is overconfidence, as exemplified in Fig. 3b, c. An *overconfident* classifier thinks it's better at separating classes than it actually is. Hence we need to make predicted probabilities less extreme by pushing them toward the centre. The paper we took this figure from Niculescu-Mizil and Caruana (2005) also fitted a logistic curve to this reliability diagram Fig. 3b, because they were investigating the effect of Platt scaling in a wide range of scenarios. But we see clearly that the fit is rather poor, and using this logistic curve as a calibration map is likely to make matters worse. In this particular case we would be looking for something that can produce an inverse sigmoidal curve. What would be a principled way to achieve this?

One approach that works well is to model the classwise scores with Beta distributions (Kull et al., 2017a, 2017b). These assume random variables bounded in [0, 1] which is exactly what we need for post-hoc calibration. Beta distributions can easily model the skewed distributions resulting from overconfident classifiers, as demonstrated in Fig. 3b. Mathematically, these curves have three parameters:

$$\mathsf{c}(s;a,b,c) = \frac{1}{1+\exp(-a\ln s + b\ln(1-s) - c)}$$

This additional degree of freedom allows Beta calibration to produce both sigmoidal and inverse-sigmoidal calibration maps. It can even fit the identity map ($a = b = 1$, $c = 0$), which means that Beta calibration can recognise that scores are already calibrated. In contrast, applying logistic calibration to a calibrated classifier will decalibrate it. While Beta calibration can achieve many shapes of calibration maps, it is still a limited parametric family. Methods to fit richer calibration map families will be discussed in Sect. 5.

2.3 Multiclass calibration

The examples we have seen so far all relate to binary classification. Is it straightforward to generalise calibration concepts and methods to more than two classes? The answer to this question is somewhat involved. Some ideas do generalise quite easily: for example, we can model per-class scores by multivariate distributions such as Dirichlet distributions (Kull et al., 2019). Some other ideas don't generalise easily at all: for example, a multiclass generalisation of isotonic regression is not straightforward because rankings are inherently bipartite. Such approaches therefore need to be approximated by considering classes in a pairwise or one-versus-rest manner.

This helps to explain why there is divergence even at a definitional level. There are at least three different ways of *defining* what it means to be calibrated in a multiclass setting. They are equivalent for binary classification but increasingly stronger for more than two classes,[2] and can be summarised as follows:

- Confidence calibration: only consider the highest predicted probability.
- Classwise calibration: only consider marginal probabilities.
- Multiclass calibration: consider the entire vector of predicted probabilities.

We will use this terminology throughout the paper to avoid confusion.

Confidence calibration was proposed by Guo et al. (2017), although not under that name. It requires that among all instances where the probability of *the most likely class* is predicted to be α, the expected accuracy is α. Let $\mathbb{X}$ denote the instance space and $\mathbb{P}_{\mathbb{Y}}$ the probability K-simplex, then a probabilistic classifier $\mathsf{f} : \mathbb{X} \to \mathbb{P}_{\mathbb{Y}}$ is *confidence-calibrated*, if for any confidence level $\alpha \in [0, 1]$, the actual proportion of the predicted class, among all possible instances $\mathbf{x}$ being predicted this class with confidence α, is equal to α:

$$\mathsf{P}(Y = \hat{Y} \mid S_{\hat{Y}} = \alpha) = \alpha \qquad \text{where } \hat{Y} = \operatorname*{argmax}_j S_j.$$

Here, S_j denotes the j-th dimension of $\mathsf{f}(\mathbf{X})$.

Classwise calibration, proposed by Zadrozny and Elkan (2002), requires that all *one-vs-rest* probability estimators obtained from the original multiclass model are calibrated. Formally, a probabilistic classifier $\mathsf{f} : \mathbb{X} \to \mathbb{P}_{\mathbb{Y}}$ is *classwise-calibrated*, if for any class $i \in \{1, \ldots, K\}$ and any predicted probability value $s \in [0, 1]$ for this class, the actual proportion of class i, among all possible instances on the random variable $\mathbf{X}$ getting the same prediction on class i, is equal to s:

$$\mathsf{P}(Y = i \mid S_i = s) = s$$

Multiclass calibration, also called *calibration in the strong sense* (Widmann et al., 2019), is the strongest form of calibration for more than two classes, subsuming the previous two definitions. A probabilistic classifier $\mathsf{f} : \mathbb{X} \to \mathbb{P}_{\mathbb{Y}}$ is *multiclass-calibrated* if for any prediction vector $\mathbf{s} = [s_1, \ldots, s_K] \in \mathbb{P}_{\mathbb{Y}}$, the proportions of classes among all possible instances on the random variable $\mathbf{X}$ getting the same prediction $\mathsf{f}(\mathbf{X}) = \mathbf{s}$ are equal to the prediction vector $\mathbf{s}$:

$$\mathsf{P}(Y = i \mid \mathsf{f}(\mathbf{X}) = \mathbf{s}) = s_i \qquad \text{for } i = 1, \ldots, K.$$

For practical purposes, the conditions in these definitions need to be relaxed. This is where binning comes in. Once we define the bins, we can draw a reliability diagram as in the

[2] In rare cases it is possible for a multiclass classifier to be classwise calibrated but not confidence calibrated. Here is an example with 40 instances in four groups: 10 instances with predicted probabilities (0.3,0.3,0.4) and actual class distribution (4,2,4); another 10 with probabilities (0.4,0.3,0.3) and classes (3,4,3); another 10 with probabilities (0.4,0.6,0.0) and classes (5,5,0); and the remaining 10 instances with probabilities (0.3,0.6,0.1) and classes (2,7,1). This classifier is classwise-calibrated but not confidence-calibrated. There are 20 instances with confidence level 0.6 and accuracy 0.6 (5+7=12 out of 20), but the remaining 20 instances have confidence 0.4 and accuracy 0.35 (4+3=7 out of 20). The key idea of this construction is that classwise calibration considers the instances with 0.4 predicted for class 1 as a single group, but confidence calibration looks among them only at those instances for which 0.4 is the highest probability – e.g., it does consider (0.4,0.3,0.3) but not (0.4,0.6,0.0).

two-class case. For classwise calibration, we can show per-class reliability diagrams or a single averaged one. One way to assess the degree of calibration is by means of the gaps in the reliability diagram. These topics will be explored in Sect. 4. We will first take a closer look at the important concept of *proper scoring rules* in the next section. Proper scoring rules are instance-wise evaluation measures for probability estimates that avoid the need for score binning. As a result they cannot measure the degree of calibration directly, but through decomposition results, we will show how the scores can be decomposed into various components, including the calibration loss.

3 From proper scoring rules to calibration

As early as 1950 researchers were interested in scoring functions that would evaluate probabilistic forecasts without influencing the forecaster into making undesirable predictions (Brier, 1950), thus keeping the forecaster "honest" while encouraging useful forecasts. In the late 1960s the term *proper scoring rule* was already in use for score functions whose computation was based on a forecaster's probabilities and the actual occurrence or not of the corresponding events (Winkler, 1969). Informally, a scoring rule is said to be proper if optimal values are obtained by the forecaster predicting the true probabilities of the events. Proper scoring rules and their ability to induce useful and accurate forecasts also lend themselves to game-theoretic interpretations, as discussed in the seminal work of Savage (1971).

Before diving into details, let us introduce the notation. In a multiclass classification task with K classes, let the true class of an instance be a vector $\dot{\mathbf{y}} = [y_1, \ldots, y_K]$, where $y_j = 1$ when the true class is j, otherwise $y_j = 0$. Additionally, let $\mathbf{s} = [s_1, \ldots, s_K]$ be a class probability vector obtained from a classification model for an instance, i.e. $s_j \geq 0$, $j = 1, \ldots, K$ and $\sum_{j=1}^{K} s_j = 1$. A scoring rule $\phi(\mathbf{s}, \dot{\mathbf{y}})$ is a non-negative measure of how well the estimated probability vector $\mathbf{s}$ matches the true class vector $\dot{\mathbf{y}}$. We assume that lower values of $\phi(\mathbf{s}, \dot{\mathbf{y}})$ are better.[3]

Popular scoring rules include Brier score ϕ_{BS} and log-loss ϕ_{LL}, which are defined as follows:

$$\phi_{\mathrm{BS}}(\mathbf{s}, \dot{\mathbf{y}}) = \sum_{j=1}^{K} (s_j - y_j)^2. \tag{1}$$

$$\phi_{\mathrm{LL}}(\mathbf{s}, \dot{\mathbf{y}}) = -\sum_{j=1}^{K} y_j \log s_j. \tag{2}$$

Our Brier score definition ranges between 0 and 2, agreeing with Brier's original definition (Brier, 1950), although it is now common for half this quantity to be called Brier score. While scoring rules evaluate the loss incurred by the class probability estimates for a single instance, we are usually interested in evaluating the performance of a model on test data.

[3] Some authors (Winkler, 1969; Gneiting & Raftery, 2007) use scoring rules in the opposite way, i.e. higher is better, sometimes calling them utility or payoff functions, in the sense that a forecaster is rewarded for giving better predictions. If a scoring rule was proposed as a payoff function, we can typically negate its values such that lower values are better.

Thus, given a test dataset, we can calculate the empirical loss as the average instance-wise loss across the data.

Suppose the class labels $\dot{\mathbf{y}}$ are drawn from the true distribution $\mathbf{q}$ over classes. A scoring rule ϕ is proper if its expected value on probability vector $\mathbf{s}$ and $\dot{\mathbf{y}}$ is higher than (i.e., worse) or the same as its expected value on $\mathbf{q}$ and $\dot{\mathbf{y}}$, that is if

$$\mathsf{E}_{\dot{\mathbf{y}}\sim\mathbf{q}}\left[\phi(\mathbf{s},\dot{\mathbf{y}})\right] \geq \mathsf{E}_{\dot{\mathbf{y}}\sim\mathbf{q}}\left[\phi(\mathbf{q},\dot{\mathbf{y}})\right],$$

and it is strictly proper if and only if both sides of the expression being equal implies $\mathbf{s} = \mathbf{q}$. Both Brier score and log-loss are strictly proper scoring rules.

Among proper scoring rules, log-loss in particular is frequently used as the training loss of machine learning methods such as neural networks, often under the name of cross-entropy. It is also the loss function used by most of the calibration methods discussed in Sect. 5, including Platt scaling, temperature scaling and its variations (vector and matrix), beta calibration and Dirichlet calibration. A notable exception is isotonic calibration, which optimised Brier score.

At the instance level, log-loss only depends on the predicted class probability for the true class (as all other classes y_j are zero) and it might not be obvious why it is a proper scoring function. For example, one could argue that it does not matter whether the model predicts [0.8, 0.2, 0.0] or [0.8, 0.1, 0.1] on an instance of class 1, as log-loss would be $-\ln(0.8)$ in both cases. However, one needs to keep in mind that properness is defined as an expectation over a population. For instance, suppose that, for a population of instances, the true probability is [0.8, 0.2, 0.0]. In this case, the expected log-loss will be $-(0.8\log(s_1) + 0.2\log(s_2))$, which can only be minimised by predicting the probability vector [0.8, 0.2, 0.0], satisfying the properness condition. Note that log-loss can be dominated by cases where the true class is predicted with low probability: in the extreme case of a single instance with zero probability for the true class the loss is infinite.

3.1 Decompositions of proper scoring rules

When proposing the forecast evaluation that would later be known as Brier score, Brier mentioned that it could not be "gamed", i.e. in order to obtain low scores, the forecaster has to produce honest, useful and correct predictions. Later research focused on why this was true for Brier score and other proper scoring rules.

Sanders (1963) showed that Brier score corresponds to the sum of two aspects of probability evaluation: validity and sharpness, terms introduced by Miller (1962) and Bross (1954), respectively. Validity, later known as reliability (Murphy & Winkler, 1977), is now often called *calibration loss* (Kull & Flach, 2015). It refers to the fit between the forecasts and the frequency of occurrence of the event, which means that it is a joint property of the predictions and the events. Sharpness, which is now commonly called *refinement loss* and was also known as resolution (Murphy & Winkler, 1977), is the loss due to predicting the same probability for instances from different classes. DeGroot and Fienberg (1983) showed that these two terms make up proper scoring rules in general, and that probabilistic predictions should minimise refinement loss subject to calibration.

We now formally introduce such proper scoring rule decompositions. Let $\mathbf{X}$ and $\dot{\mathbf{Y}} = [Y_1, \ldots, Y_K]$ be multivariate random variables, respectively corresponding to the features and the class of a randomly picked instance, where $Y_j = 1$ if the instance is of class j,

Table 2 Example dataset with three classes and three features. Column $Q^{(i)}$ shows class posteriors given by the optimal model with $Q_1 = 0.25X_1$, $Q_2 = 0.25X_2$ and $Q_3 = 1 - (Q_1 + Q_2)$. Column $S^{(i)}$ corresponds to estimated class posteriors $S = f(X)$, which are given by $S_1 = 0.3X_3$, $S_2 = 0.1X_2$ and $S_3 = 1 - (S_1 + S_2)$. Finally, column C represents calibrated probabilities

i	$X^{(i)}$	$Y^{(i)}$	$Q^{(i)}$	$S^{(i)}$	$C^{(i)}$
1	(3, 1, 3)	(1, 0, 0)	(0.75, 0.25, 0.0)	(0.9, 0.1, 0.0)	(0.75, 0.25, 0.0)
2	(3, 1, 3)	(1, 0, 0)	(0.75, 0.25, 0.0)	(0.9, 0.1, 0.0)	(0.75, 0.25, 0.0)
3	(3, 1, 3)	(1, 0, 0)	(0.75, 0.25, 0.0)	(0.9, 0.1, 0.0)	(0.75, 0.25, 0.0)
4	(3, 1, 3)	(0, 1, 0)	(0.75, 0.25, 0.0)	(0.9, 0.1, 0.0)	(0.75, 0.25, 0.0)
5	(2, 2, 2)	(1, 0, 0)	(0.50, 0.50, 0.0)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)
6	(2, 2, 2)	(0, 1, 0)	(0.50, 0.50, 0.0)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)
7	(0, 2, 2)	(0, 1, 0)	(0.0, 0.50, 0.50)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)
8	(0, 2, 2)	(0, 1, 0)	(0.0, 0.50, 0.50)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)
9	(0, 2, 2)	(0, 0, 1)	(0.0, 0.50, 0.50)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)
10	(0, 2, 2)	(0, 0, 1)	(0.0, 0.50, 0.50)	(0.6, 0.2, 0.2)	(0.17, 0.50, 0.33)

otherwise $Y_j = 0$, for $j = 1, 2, \ldots, k$. Given a scoring classifier, class probability estimator or probability forecaster f, we denote by $\mathbf{S} = [S_1, S_2, ..., S_k] = \mathsf{f}(\mathbf{X})$ the vector output of f for random feature $\mathbf{X}$, where $\mathbf{S}$ is a random vector due to its dependence on $\mathbf{X}$. The expected value of proper scoring rule ϕ with respect to $\mathbf{S}$ and $\dot{\mathbf{Y}}$ is then $\mathsf{E}\left[\phi\left(\mathbf{S}, \dot{\mathbf{Y}}\right)\right]$.

Consider a classification test set of 10 instances belonging to three classes ($K = 3$) and described by 3 features, as shown in column $\mathbf{X}^{(i)}$ of Table 2. Suppose instances 1, 2, 3 and 5 belong to class 1, i.e. $Y_1 = 1$, instances 4, 6, 7, 8 belong to class 2 and the rest belong to class 3. Note that in this example, the optimal class posteriors $\mathbf{Q}^{(i)}$ are given, however this is not generally the case. Additionally, the sample is representative by design, with average $\mathbf{Y}^{(i)}$, among instances with the same feature values, exactly agreeing with $\mathbf{Q}^{(i)}$. Given columns $\mathbf{S}^{(i)}$ and $\mathbf{Y}^{(i)}$ from Table 2, and using Equations (1) and (2), by averaging over the 10 instances, we get a mean Brier score of 0.71 and mean log-loss of 1.12.

Now let $\mathbf{C} = [C_1, C_2, \ldots, C_K]$ be yet another random vector, where $C_j = \mathsf{E}[Y_j|\mathbf{S}]$, i.e. C_j represents the actual proportion of class j given all instances which received the same estimate $\mathbf{S}$ from model f. The decomposition of proper scoring rule ϕ into calibration loss and refinement loss is then the sum of expected loss between $\mathbf{S}$ and $\mathbf{C}$ and expected loss between $\mathbf{C}$ and $\mathbf{Y}$ (Bröcker & Smith, 2007; Kull & Flach, 2015):

$$\mathsf{E}\left[\mathsf{d}\left(\mathbf{S}, \dot{\mathbf{Y}}\right)\right] = \mathsf{E}\left[\mathsf{d}\left(\mathbf{S}, \mathbf{C}\right)\right] + \mathsf{E}\left[\mathsf{d}\left(\mathbf{C}, \dot{\mathbf{Y}}\right)\right],$$

where $\mathsf{d} : \mathbb{P}_{\mathbb{Y}} \times \mathbb{P}_{\mathbb{Y}} \to \mathbb{R}$ is the divergence associated to proper scoring rule ϕ. For log-loss and Brier score, d has been shown to correspond to the Kullback-Leibler divergence and the mean squared difference, respectively (Kull & Flach, 2015).

The definition above is given based on expected population loss. In practice, we would be calculating these quantities on test data, as we generally do not have access to the whole population. Thus, we actually calculate an empirical loss. But assuming that test data is representative of the whole population and that there is a uniform distribution over the test instances and zero probability elsewhere, empirical loss can be interpreted as a special case of expected loss and all decompositions derived for expected loss also apply to empirical loss (Kull & Flach, 2015).

In our example in Table 2, the first four instances share the same model scores $\mathbf{S} = [0.9, 0.1, 0.0]$, with the first three instances belonging to class 1 and the fourth to class 2. Thus, for these instances, $\mathbf{C} = [0.75, 0.25, 0.0]$. The other six instances also share a score vector, with 1 instance from class 1, three from class 2 and 2 instances from class 3. As a result, they share the same calibrated probabilities $\mathbf{C} = [1/6, 3/6, 2/6]$. Given these values and considering Brier score, calibration loss is 0.19 and refinement loss is 0.52, which sum up to the Brier score of 0.71, as expected. For log-loss the decomposition is $1.12 = 0.29 + 0.83$.

It is easy to see that a function that maps $\mathbf{S}$ to $\mathbf{C}$, known as a calibrator in Sect. 5, could potentially reduce calibration loss to 0, consequently decreasing proper scoring rule loss. Refinement loss, on the other hand, is only zero if there are no ties between instances of different classes, i.e. they do not share the same value for $\mathbf{S}$. If that is the case, it is possible to map $\mathbf{S}$ to $\mathbf{C}$ such that it is perfectly confident and correct, i.e. $C_j = Y_j$.

Now suppose we knew the optimal model that outputs the true posterior class probabilities $\mathbf{Q} = [Q_1, Q_2, \ldots, Q_K]$, where $Q_j = \mathsf{E}[Y_j|\mathbf{X}]$, i.e. Q_j is the true proportion of class j among instances with feature values $\mathbf{X}$. Then the expected loss according to proper scoring rule ϕ can be decomposed into the sum of expected divergences between $\mathbf{S}$ and $\mathbf{Q}$ and $\mathbf{Q}$ and $\mathbf{Y}$:

$$\mathsf{E}\Big[\mathsf{d}\big(\mathbf{S}, \dot{\mathbf{Y}}\big)\Big] = \mathsf{E}\Big[\mathsf{d}\big(\mathbf{S}, \mathbf{Q}\big)\Big] + \mathsf{E}\Big[\mathsf{d}\big(\mathbf{Q}, \dot{\mathbf{Y}}\big)\Big],$$

The first term in the sum is called epistemic loss (Senge et al., 2014) and is the loss due to our model $\mathsf{f}(\mathbf{X})$ not being optimal. The second term is called irreducible or aleatoric loss (Senge et al., 2014) and is the loss of the optimal model. Going back to the example in Table 2, for Brier score, epistemic loss is 0.26 and irreducible loss is 0.45. For log-loss the decomposition is $1.12 = 0.48 + 0.64$. The irreducible loss is only zero if the attributes of every instance $\mathbf{X}$ carry enough information to uniquely define the label $\dot{\mathbf{Y}}$. It is thus not possible to come up with a procedure to reduce the overall loss based on it. Additionally it would be unrealistic to do so based on epistemic loss. We are then naturally drawn to calibration loss as it is simple to interpret (we want our predicted probabilities of occurrence of an event to match the actual frequencies of occurrence) and also because we can easily estimate it with enough data. Thus, as discussed in the following section, visualisation and estimation of miscalibration have received considerable attention in the literature of probability evaluation and machine learning.

4 Evaluation and visualisation of classifier calibration

In this section we present a range of approaches to evaluate and visualise calibration, and discuss the advantages and disadvantages of each. In Sect. 4.1 we explore the first steps towards measuring miscalibration, starting from the early work on weather forecast calibration, and discuss various approaches to visually inspect miscalibration. Some of those early approaches have been later adapted and proposed as new calibration metrics which are discussed in the following sections. We start with binary calibration in Sect. 4.2 which will serve as a building block for the two multiclass calibration measures discussed in Sects. 4.3 and 4.4.

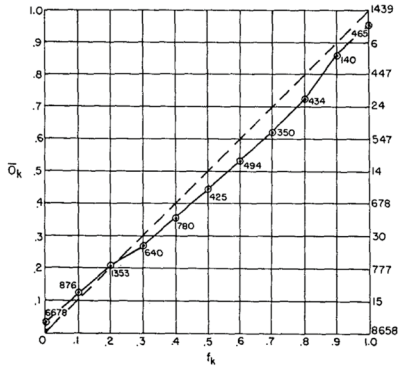
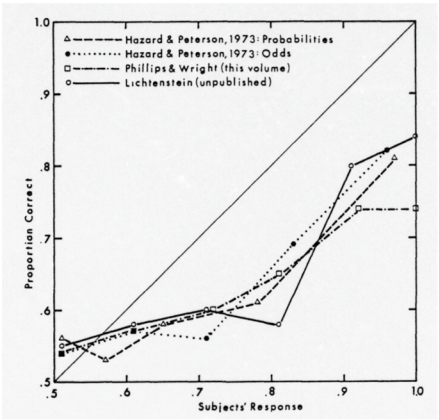
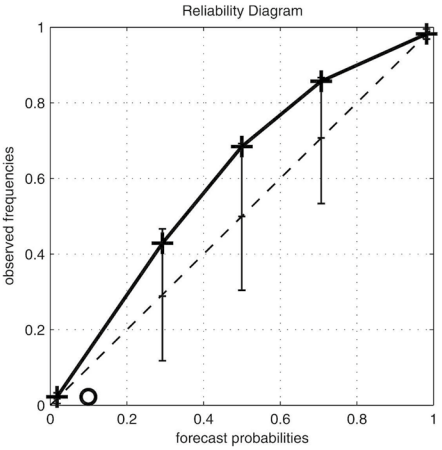


FIG. 1. Forecast probability, f_k, vs. observed relative frequency of occurrence, O_k, for instructors' forecasts in 1955–1956 seasons.

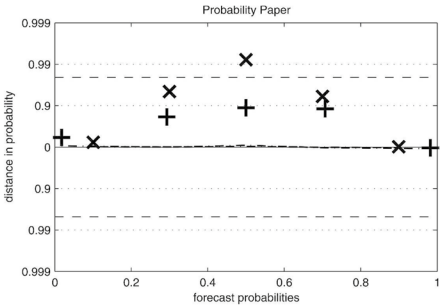
(a) (Sanders, 1963, Fig. 1)



(b) (Lichtenstein et al., 1977, Fig. 2)



(c) (Bröcker & Smith, 2007, Fig. 1)



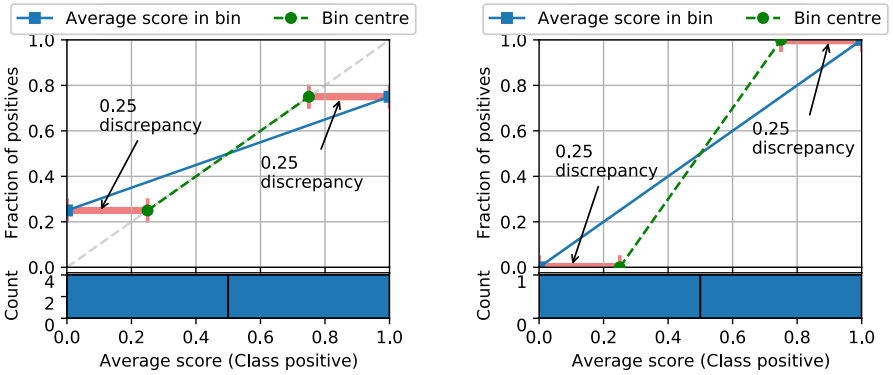
(d) (Bröcker & Smith, 2007, Fig. 4)

Fig. 5 Different graphical representations of calibration-related information found in the literature

4.1 Visualising calibration

Several visualisation techniques have been proposed for easier inspection of the reliability of predictions at different score ranges. In this section we summarise a range of approaches that have been proposed in the literature, from simple tabular inspection to the more recent reliability diagrams (see Bröcker and Smith (2007) for an extended discussion of reliability diagrams).

Starting from the binary case, the basic idea has been to show the relation between a particular predicted score $\hat{s} \in [0, 1]$ for the positive class of a probabilistic classifier and the respective observed proportion of positives, which we denote by $\bar{y}(\hat{s})$. In order to compute a observed proportion of positives we require multiple samples with the same score $\hat{s}$. However, given that the scores are continuous variables, the probability of getting multiple samples with the same score is almost zero. For that reason, the score space is commonly discretised into M bins $\{\mathbb{B}_1, \ldots, \mathbb{B}_M\}$, thus increasing the probability of having multiple



(a) Not calibrated example. 4 samples with score 0 from which 25% are positive, and 4 samples with score 1 from which 75% are positive.

(b) Calibrated example. 1 sample with score 0 from negative class, and 1 sample with score 1 from positive class.

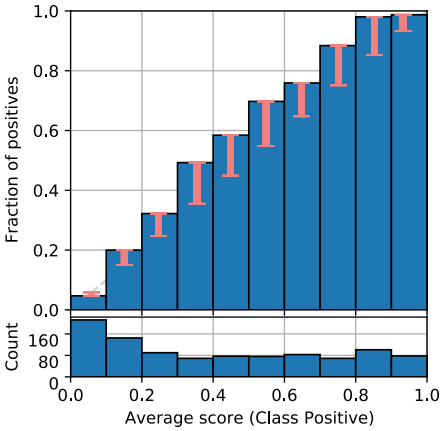
Fig. 6 Simple example showing the maximum discrepancy in calibration error when using the centre of the bin for the marker instead of the average score of the bin. Both examples show only two bins [0, 0.5) and [0.5, 1]. **a** shows a model not calibrated, but using the centre of the bin (green dashed line) visually seems calibrated. The gap between the centre of the bin and the average score is maximum, which corresponds to half of the bin size with respect to the average score (red line). **b** shows a calibrated model which seems uncalibrated if using the centre of the bin. The gap is again half of the bin size

samples per estimation range (perhaps by forcing each bin to have a minimum number of samples). Then, we can obtain M observed proportions of positives $\bar{y}(\mathbb{B}_m)$ instead of one per score value $\hat{s}$. See Table 1 for an early use of observed proportions of positives using equal-width bins of predicted probabilities.

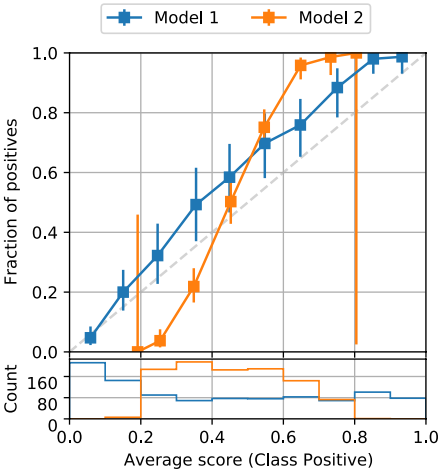
Later work showed line plots instead of tabular data, with markers at the centre of each bin (Sanders, 1963; Lichtenstein et al., 1977, 1982; Murphy & Winkler, 1977); occasionally including the number of samples in each bin next to the markers (See Fig. 5a). Other authors showed the markers with size relative to the sample size (Hagedorn et al., 2005), or as a separate histogram next to the reliability diagram (Niculescu-Mizil & Caruana, 2005). Line plots facilitate visual inspection of the discrepancy of the observed proportions of positives against a perfectly calibrated model, which corresponds to the diagonal. However, positioning the markers at the centre of the bins can be misleading, as even perfectly calibrated scores may result in a *visually* non-calibrated reliability diagram with a deviation of up to half the bin width from the diagonal (Bröcker & Smith, 2007) (See Fig. 6 for an example). This may happen in cases in which the scores are not evenly distributed in each bin.

In order to alleviate this problem, later work centred the markers on the average predicted scores $\bar{s}(\mathbb{B}_m)$ in each bin instead.

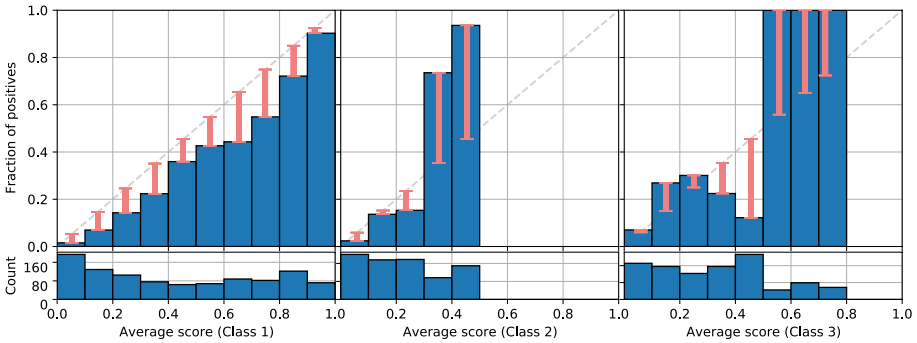
There are two main methods to group the predicted scores into bins to build reliability diagrams. The first one, commonly called *width binning*, distributes scores into a number of bins of the same width, choosing bin edges accordingly (e.g., 10 bins $[0, 0.1], (0.1, 0.2], \ldots,$ (0.9, 1.0]). Another method, called *frequency binning* (also equal size or equal mass binning) groups the scores by keeping the same number of samples in each bin. Frequency binning can and will normally result in bins with different widths, but it ensures that all bins have the same "weight" when assessing the model's miscalibration.



(a) Reliability diagram with bars for each bin and red error gaps with regards to a perfectly calibrated model, and a histogram of the scores at the bottom.



(b) Comparison of multiple calibration methods, with Clopper–Pearson binomial confidence intervals of 95% in each observed proportion of positives.



(c) Multiple classes can be shown as one-vs-rest reliability diagrams.

Fig. 7 Examples of reliability diagrams using the Python library PyCalib

Bin estimations have varying reliability, e.g., depending on the number of samples in the bin. One way to show this visually is by incorporating confidence intervals. One of the simplest methods is to assume that each bin is sampled from a binomial distribution, and compute the corresponding confidence intervals of the observed proportion of positives (see Fig. 7b). However, making the assumption of fixed bins, score averages and proportion of positives is unrealistic; unless the score distribution is uniform (see Bröcker and Smith (2007) for a detailed discussion). More accurate intervals can be obtained by using the *consistency resampling* method proposed by Bröcker and Smith (2007), which consists on sampling the scores with replacement several times in order to obtain multiple surrogate forecasts and observations and their respective quantiles.

More recently, binary reliability diagrams have been extended to the multiclass setting. One method to show multiclass scores in a binary fashion consists in plotting only the *confidence* of the model for each sample (the highest score among all possible classes). This

Table 3 Class probabilities given by a classifier to 30 instances belonging to three different classes

id	$\hat{s}_1$	$\hat{s}_2$	$\hat{s}_3$	y
1	1.0	0.0	0.0	1
2	0.9	0.1	0.0	1
3	0.8	0.1	0.1	1
4	0.7	0.1	0.2	1
5	0.6	0.3	0.1	1
6	0.4	0.1	0.5	1
7	1/3	1/3	1/3	1
8	1/3	1/3	1/3	1
9	0.2	0.4	0.4	1
10	0.1	0.5	0.4	1
11	0.8	0.2	0.0	2
12	0.7	0.0	0.3	2
13	0.5	0.2	0.3	2
14	0.4	0.4	0.2	2
15	0.4	0.2	0.4	2
16	0.3	0.4	0.3	2
17	0.2	0.3	0.5	2
18	0.1	0.6	0.3	2
19	0.1	0.3	0.6	2
20	0.0	0.2	0.8	2
21	0.8	0.2	0.0	3
22	0.8	0.1	0.1	3
23	0.8	0.0	0.2	3
24	0.6	0.0	0.4	3
25	0.3	0.0	0.7	3
26	0.2	0.6	0.2	3
27	0.2	0.4	0.4	3
28	0.0	0.4	0.6	3
29	0.0	0.3	0.7	3
30	0.0	0.3	0.7	3

approach was already used for binary classification by Lichtenstein et al. (1977), reproduced here as Fig. 5b. The figure shows a binary reliability diagram with scores in the range [0.5, 1] (with K classes the confidence is never lower than $1/K$). Another example of the use of confidences in multiclass problems is the recent work of Guo et al. (2017), which shows the confidence for multiclass problems in the form of bar plots and the ECE and MCE metrics discussed below. Finally, instead of collapsing all classes into a single diagram one could construct one reliability diagram per class as line plots or bars in a one-vs-rest manner (Kull et al., 2019) (see Fig. 7c).

4.2 Evaluating calibration error: the binary case

We now proceed with discussing how to measure calibration error. We will use the three-class toy problem in Table 3 throughout. In this section we convert it to a binary

Table 4 Positive probabilities and corresponding instance labels separated into 5 bins. The probabilities were obtained by taking class 1 as the positive class in the multiclass problem in Table 3. To reduce the width of the table, a number followed by a superscript parenthesis $X^{(Y)}$ indicates a list of number X repeated Y times

$\mathbb{B}_m$	$\lvert\mathbb{B}_m\rvert$	$\mathsf{s}(\mathbb{B}_m)$	$\bar{\mathsf{s}}(\mathbb{B}_m)$	$\mathsf{y}(\mathbb{B}_m)$	$\bar{\mathsf{y}}(\mathbb{B}_m)$	$\lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{\mathsf{s}}(\mathbb{B}_m)\rvert$
$\mathbb{B}_1$	11	$0.0^{(4)}, 0.1^{(3)}, 0.2^{(4)}$	0.10	$0^{(9)}, 1^{(2)}$	0.18	0.08
$\mathbb{B}_2$	7	$0.3^{(2)}, 1/3^{(2)}, 0.4^{(3)}$	0.35	$0^{(4)}, 1^{(3)}$	0.43	0.08
$\mathbb{B}_3$	3	0.5, 0.6, 0.6	0.57	0, 0, 1	0.33	0.24
$\mathbb{B}_4$	7	$0.7^{(2)}, 0.8^{(5)}$	0.77	$0^{(5)}, 1^{(2)}$	0.29	0.48
$\mathbb{B}_5$	2	0.9, 1.0	0.95	1, 1	1.00	0.05

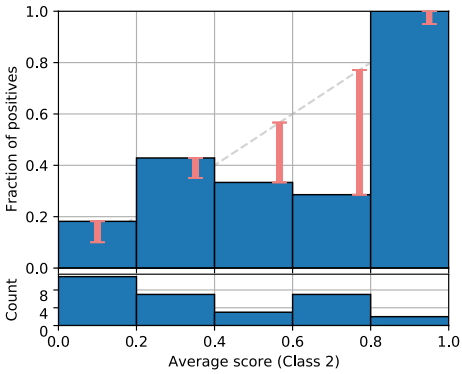


Fig. 8 Reliability diagram corresponding to the 5 bins presented in Table 4

classification problem by taking class 1 as positive and classes 2 and 3 as negative. We then apply equal-width binning to the class 1 probabilities. Table 4 shows the resulting bins $\mathbb{B}_m$, $m \in \{1, \ldots, 5\}$, from which we calculate the number of instances in the bin denoted as $\lvert\mathbb{B}_m\rvert$, the average probability $\bar{\mathsf{s}}(\mathbb{B}_m)$, the proportion of positives $\bar{\mathsf{y}}(\mathbb{B}_m)$, and the absolute gap between them $\lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{\mathsf{s}}(\mathbb{B}_m)\rvert$.

The values of $\bar{\mathsf{p}}(\mathbb{B}_m)$, $\bar{\mathsf{y}}(\mathbb{B}_m)$ and $\lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{\mathsf{s}}(\mathbb{B}_m)\rvert$ in Table 4 can be used to draw the reliability diagram shown in Fig. 8, where we can see the red bars representing the calibration error at each bin.

Given the information about the bins in Table 4 we could calculate an average of the bin-wise calibration error, which leads us to binary estimated calibration error, as defined in Definition 1.

Definition 1 (*Binary-ECE*) Binary *estimated calibration error* (Roelofs et al., 2021), or binary-ECE (Naeini et al., 2015), is the average gap across all bins in a reliability diagram, weighted by the number of instances in each bin:

$$\mathsf{ECE}_{\text{binary}} = \sum_{m=1}^{M} \frac{\lvert\mathbb{B}_m\rvert}{N} \lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{\mathsf{s}}(\mathbb{B}_m)\rvert,$$

where M and N are the number of bins and instances, respectively. Binary-ECE is also commonly called binary *expected calibration error*, however this quantity is empirically

estimated on a validation or test sample, which is why we prefer to call it binary *estimated calibration error*.

Binary-ECE aggregates part of the visual information about calibration error in Fig. 8 by taking the weighted mean of the sizes of these gaps. In this example, $\mathsf{ECE}_{\text{binary}} = 0.1873$.

Binary-ECE ranges from 0 to 1, with particular values including $\mathsf{ECE}_{\text{binary}} = \pi_1$ (proportion of positives) for a classifier that randomly outputs probability 1 for one of the classes, $\mathsf{ECE}_{\text{binary}} = 0$ for a perfect classifier that always predicts probability 1 for the correct class or for a classifier that constantly outputs positive probability equal to π_1, and $\mathsf{ECE}_{\text{binary}} = 1$ for a classifier that always assigns probability 1 to the wrong class.

Instead of measuring how miscalibrated the model is on average, one might be interested in the worst case, in other words "how maximally miscalibrated is the model"? In this case, one might use the binary maximum calibration error (Naeini et al., 2015).

Definition 2 (*Binary-MCE*) Binary maximum calibration error (binary-MCE) is the maximum gap across all bins in a reliability diagram:

$$\mathsf{MCE}_{\text{binary}} = \max_{m \in \{1,\ldots,M\}} |\bar{\mathrm{y}}(\mathbb{B}_m) - \bar{\mathrm{s}}(\mathbb{B}_m)|.$$

In our example, Table 4 and Fig. 8 show that the fourth bin has the largest gap, with its size corresponding to $\mathsf{MCE}_{\text{binary}} = 0.48$.

Binary-MCE can be very sensitive to bins with few instances, where the values of $\bar{\mathrm{y}}(\mathbb{B}_m)$ and $\bar{\mathrm{s}}(\mathbb{B}_m)$ are calculated based on very little information. In extreme cases with only a single instance in a bin, $\bar{\mathrm{y}}(\mathbb{B}_m)$ will be the instance's actual label (1 or 0), potentially resulting in a large gap that would have a small weight in binary-ECE, while being the whole value of binary-MCE, meaning that the model would be evaluated based on its performance for a single instance.

4.3 Classwise calibration error

We now proceed to settings with more than two classes and start by defining the estimated calibration error corresponding to a single class j in a multiclass task.

Definition 3 (*Class-j-ECE*) Given a multiclass task with K classes, class-j-ECE corresponds to the binary-ECE calculated by taking class j as positive and the other $K-1$ classes together as a new negative class:

$$\mathsf{ECE}_j = \sum_{m=1}^{M} \frac{|\mathbb{B}_{m,j}|}{N} |\bar{\mathrm{y}}_j(\mathbb{B}_{m,j}) - \bar{\mathrm{s}}_j(\mathbb{B}_{m,j})|,$$

where $\mathbb{B}_{m,j}$ is the m-th bin of class j.

Once again we turn to our example in Table 3. Following Definition 3, we have already calculated class-1-ECE (0.1873) in our binary-ECE example. Now we need to calculate class-2-ECE and class-3-ECE. Tables 5 and 6 show the binary bins for classes 2 and 3, respectively, while the binary reliability diagrams of the three classes can be seen in Fig. 9.

Table 5 Positive probabilities and corresponding instance labels separated into 5 bins. The probabilities were obtained by taking class 2 as the positive class in the multiclass problem in Table 3

$\mathbb{B}_{m,2}$	$\lvert\mathbb{B}_{m,2}\rvert$	$\mathsf{s}(\mathbb{B}_{m,2})$	$\bar{\mathsf{s}}(\mathbb{B}_{m,2})$	$\mathsf{y}(\mathbb{B}_{m,2})$	$\bar{\mathsf{y}}(\mathbb{B}_{m,2})$	$\lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{\mathsf{s}}(\mathbb{B}_m)\rvert$
$\mathbb{B}_{1,2}$	15	$0^{(5)}, 0.1^{(5)}, 0.2^{(5)}$	0.10	$0^{(10)}, 1^{(5)}$	0.33	0.23
$\mathbb{B}_{2,2}$	12	$0.3^{(5)}, 1/3^{(2)}, 0.4^{(5)}$	0.35	$0^{(8)}, 1^{(4)}$	0.33	0.02
$\mathbb{B}_{3,2}$	3	0.5, 0.6, 0.6	0.57	0, 0, 1	0.33	0.24
$\mathbb{B}_{4,2}$	0					
$\mathbb{B}_{5,2}$	0					

Table 6 Positive probabilities and corresponding instance labels separated into 5 bins. The probabilities were obtained by taking class 3 as the positive class in the multiclass problem in Table 3

$\mathbb{B}_{m,3}$	$\lvert\mathbb{B}_{m,3}\rvert$	$\mathsf{s}(\mathbb{B}_{m,3})$	$\bar{\mathsf{s}}(\mathbb{B}_{m,3})$	$\mathsf{y}(\mathbb{B}_{m,3})$	$\bar{\mathsf{y}}(\mathbb{B}_{m,3})$	$\lvert\bar{\mathsf{y}}(\mathbb{B}_m) - \bar{p}(\mathbb{B}_m)\rvert$
$\mathbb{B}_{1,3}$	11	$0.0^{(4)}, 0.1^{(3)}, 0.2^{(4)}$	0.10	$0^{(7)}, 1^{(4)}$	0.36	0.26
$\mathbb{B}_{2,3}$	11	$0.3^{(4)}, 1/3^{(2)}, 0.4^{(5)}$	0.35	$0^{(9)}, 1^{(2)}$	0.18	0.17
$\mathbb{B}_{3,3}$	4	0.5, 0.5, 0.6, 0.6	0.55	0, 0, 0, 1	0.25	0.3
$\mathbb{B}_{4,3}$	4	0.7, 0.7, 0.7, 0.8	0.72	0, 1, 1, 1	0.75	0.03
$\mathbb{B}_{5,3}$	0					

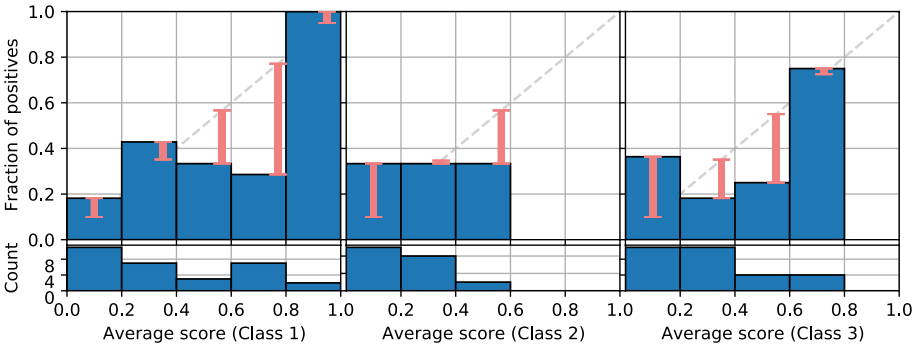


Fig. 9 Binary reliability diagram of each class in our running example. Note the empty bins in the second and third reliability diagrams, showing that the classifier never outputs high probabilities for these classes

From the values in Tables 5 and 6 and following Definition 3, we get class-2-ECE = 0.147 and class-3-ECE = 0.2017.

We can now define classwise-ECE (Kull et al., 2019).

Definition 4 (*Classwise-ECE*) The estimated classwise calibration error (classwise-ECE) is the average class-j-ECE across all classes:

$$\mathrm{ECE}_{\mathrm{classwise}} = \frac{1}{K}\sum_{j=1}^{K}\mathrm{ECE}_j.$$

Table 7 Confidence bins corresponding to the toy example from Table 3. The binary values in the fifth column indicate if the classifier correctly classified each instance

$\mathbb{B}_m$	$\lvert\mathbb{B}_m\rvert$	$\mathsf{s}(\mathbb{B}_m)$	$\bar{\mathsf{s}}(\mathbb{B}_m)$	$\mathsf{y}(\mathbb{B}_m)$	$\mathsf{acc}(\mathbb{B}_m)$	$\lvert\mathsf{acc}(\mathbb{B}_m) - \bar{\mathsf{p}}(\mathbb{B}_m)\rvert$
$\mathbb{B}_1$	0					
$\mathbb{B}_2$	7	$1/3^{(2)}, 0.4^{(5)}$	0.38	$0^{(4)}, 1^{(3)}$	0.43	0.05
$\mathbb{B}_3$	10	$0.5^{(4)}, 0.6^{(6)}$	0.56	$0^{(7)}, 1^{(3)}$	0.30	0.26
$\mathbb{B}_4$	11	$0.7^{(5)}, 0.8^{(6)}$	0.75	$0^{(6)}, 1^{(5)}$	0.45	0.3
$\mathbb{B}_5$	2	0.9, 1.0	0.95	1, 1	1.00	0.05

According to Definition 4, in our example classwise-ECE = $(0.1873 + 0.147 + 0.2017)/3 = 0.1787$.

Definition 5 (*Classwise-MCE*) The maximum classwise calibration error (classwise-MCE) (Kull et al., 2019) is defined as the maximum gap across all bins and all classwise-reliability diagrams:

$$\mathsf{MCE}_{\text{classwise}} = \max_{j\in\{1,\ldots,K\}\, m\in\{1,\ldots,M\}} |\bar{\mathsf{y}}_j(B_{m,j}) - \bar{\mathsf{s}}_j(B_{m,j})|.$$

In our running example, $\mathsf{MCE}_{\text{classwise}} = 0.48$.

Note that this is not the only way to define classwise-MCE. For example, one could be interested in the average maximum gap across classes, i.e. an average class-*j*-MCE, which in our example would be $(0.48 + 0.24 + 0.3)/3 = 0.34$. As another variation, one could take the maximum class-*j*-ECE, which would then indicate the most miscalibrated class on average. In our example, this variation corresponds to class-3-ECE (0.2017).

Although classwise-ECE and MCE measure calibration error over all classes, they do not take into account the miscalibration that arises from every possible relationship between classes. For a true multiclass-ECE, we could bin the probability vectors in simplex space and then calculate the gaps between the average probability vector and the vector of class proportions in each bin. Although this seems like the right thing to do when evaluating multiclass calibration, it might not be efficiently computable with large numbers of classes, as the number of bins can be prohibitively high and most bins would likely be empty.

Similarly to binary-ECE, a value of classwise-ECE = 0 can be obtained by a classifier that constantly outputs the class proportions. For example, if a classifier always predicts 1/3 setosa, 1/3 versicolor and 1/3 virginica for Fisher's Iris dataset, the resulting classwise-ECE will be 0.

4.4 Confidence calibration error

As already mentioned, some calibration measures only consider a model's confidence, defined as the maximum value in the predicted probability vector, i.e. the probability given to the winning class. Prediction confidence is not a new idea and has been around since at least the 1970s. Lichtenstein et al. (1977) called it the half-range method in the binary case (referred to as "two alternatives" in the paper) and it was also mentioned as one of the variations of the multiclass task, which the authors called "three or more alternatives". More recently it has been popularised by Guo et al. (2017).

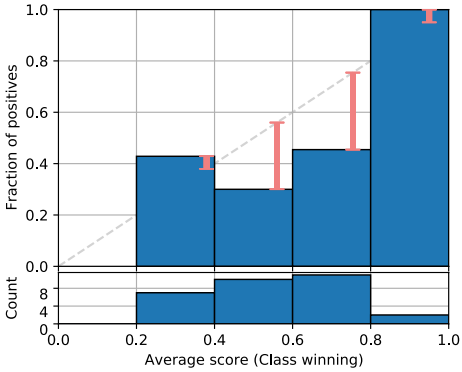


Fig. 10 Confidence reliability diagram corresponding to the bins in Table 7

To calculate confidence-ECE, we first bin the confidence values and, for each instance of each bin, we produce a binary value indicating if the classifier predicted correctly or not. Recalling the example from Table 3, the resulting confidence bins are presented in Table 7, where $\bar{\mathsf{p}}(\mathbb{B}_m)$ and $\mathsf{acc}(\mathbb{B}_m)$ are respectively the average confidence and prediction accuracy in bin $\mathbb{B}_m$, with the corresponding reliability diagram in Fig. 10. Note that bin $\mathbb{B}_1$ is empty, which demonstrates that confidence-ECE and confidence reliability diagrams are different from binary-ECE even on two-class problems.

Definition 6 (*Confidence-ECE*) Confidence-ECE is the average difference between accuracy and average confidence across all bins in a confidence reliability diagram, weighted by the number of instances in each bin:

$$\mathsf{ECE}_{\text{confidence}} = \sum_{m=1}^{M} \frac{|\mathbb{B}_m|}{N} |\mathsf{acc}(\mathbb{B}_m) - \bar{\mathsf{p}}(\mathbb{B}_m)|.$$

We then calculate confidence-ECE as the weighted mean of the gaps between average confidence and prediction accuracy at each bin, thus $\mathsf{ECE}_{\text{confidence}} = 0.2117$, while the maximum confidence calibration error (confidence-MCE) corresponds to the largest gap, i.e. $\mathsf{MCE}_{\text{confidence}} = 0.3$. As with binary-ECE and for the same reasons, confidence-MCE is very sensitive to small bins, but in this particular example, its value corresponds to the gap of the largest bin.

As we show in Sect. 5, there is a wide range of post-hoc calibration techniques in the literature, each one capable of handling different miscalibration patterns. Thus, it can be useful to first analyse a model's reliability diagram, as it allows us to diagnose model miscalibration, i.e. in what probability ranges it is underconfident or overconfident, and this can help choose an appropriate calibration method. In addition, any of the ECE measures discussed in this Section can be used to decide if a classifier actually needs to be calibrated, by testing the hypothesis that the model is already calibrated, as shown in Sect. 6. Finally, it is important to consider that all ECE/MCE measures can provide very different values given different number of bins, as shown in Figs. 1 and 2, and they can be minimised by predicting the overall class distribution, regardless of the given instance. Thus, it is good practice to use them together with the proper scoring rules presented in Sect. 3, to get a full evaluation of the probabilities produced by a model.

5 Calibration methods

In this section we provide a systematic review of standard approaches to calibrating classifiers through the construction of calibration maps. In particular, we focus on post-hoc calibration methods for classification, which can be used to adjust probability outputs separately from the initial training process. For each method, we cover the mathematical formulation, the objective function for learning the map, and the predictive function to calculate the class probabilities for a new instance. We include a few test cases to empirically analyse each method, and discuss their advantages and disadvantages. General guidance on each method's implementation is also discussed, addressing issues such as overfitting and computational effort. At the end of the section, we briefly describe some related work to classifier calibration, such as methods that can improve the calibration level during training time, methods that are designed for getting well-calibrated probability for other predictive tasks, and approaches that are designed for specific tasks in areas such as computer vision and natural language processing.

5.1 Preliminaries

We denote a calibration map as a function $\mathsf{g} : \mathbb{S}_{\mathbb{Y}} \to \mathbb{P}_{\mathbb{Y}}$. Here $\mathbb{Y} = \{1, \ldots, K\}$ is the discrete space with K classes of the target variable. $\mathbb{P}_{\mathbb{Y}}$ denotes the probability vector space on $\mathbb{Y}$, that is: $\mathbb{P}_{\mathbb{Y}} = \{[s_1, \ldots, s_K] \mid s_j \geq 0, \sum_{j=1}^{K} s_j = 1\}$. $\mathbb{S}_{\mathbb{Y}}$ represents the original output space of the (uncalibrated) classifier, which can take various forms depending on the formalisation of the classifier. For instance, for a typical probabilistic classifier, we have $\mathbb{S}_{\mathbb{Y}} = \mathbb{P}_{\mathbb{Y}}$. A one-vs-rest SVM might produce K-dimensional vectors in $\mathbb{S}_{\mathbb{Y}} = \mathbb{R}^K$. Deep neural networks can also provide unnormalised vector outputs before the softmax is performed at the final layer. Calibration maps that work with such vector spaces are referred to as scaling approaches. The name suggests these approaches are used to scale the vector output into a probability output, which is quite useful for models like SVMs, as the hinge loss does not optimise probability estimates by default. However, other than the name and historical motivation, in modern applications, probability calibrators and scaling approaches can be used interchangeably, as the probability space and vector space can be transformed into each other by using the link function (e.g. softmax) and inverse link function (e.g. logit transform).

In general, to obtain a calibration map, we will have a calibration set with uncalibrated probabilities and labels, denoted as $\mathbb{C} = \{(\mathbf{s}_1, y_1), \ldots, (\mathbf{s}_N, y_N)\}$, and an objective function $\mathsf{L}(\mathbb{C})$ to optimise. Typically, the objective function is constructed using proper scoring rules, that is, $\mathsf{L}(\mathbb{C}) = \sum_{i=1}^{N} \phi(\mathbf{s}_i, y_i)$, and it can be optimised via various gradient descent algorithms.

Training Schemes: While the above notation suggested that the calibration map can be obtained by fitting a calibrator on a given hold-out calibration set, in practice a small training set and calibration set might lead to over-fitting thus limiting the performance of the calibrated model. A typical solution here is to introduce multi-fold dataset splits as introduced in Platt (1999). The author proposes to divide the training sets into three different folds and fit the model and calibration map in cross-validation. Two folds are used to train the uncalibrated model, and the model predictions on the last fold are collected and used to fit the calibration map. This training process thus ends up with three pairs of models and calibration maps. At test time each instance will be predicted with all three pairs, and

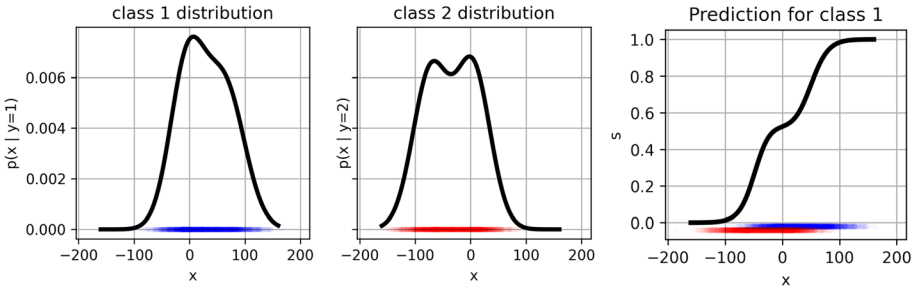


Fig. 11 The synthetic dataset used in the test cases. The left two figures shows the data distribution for each class respectively. The figure on the right gives the Bayes-optimal posterior probability for class 1

resulting probabilities are averaged to give the final estimate. Alternatively, people have also used applied label smoothing (Müller et al., 2019) to reduce the level of overfitting and over-confidence. The idea is to have a weighted average between the original probability vector (either from the uncalibrated model or after a calibration method has been applied) and a uniform probability vector. Therefore, the final probability vector will be pushed somewhat towards the uniform probability vector, which avoids overly high confidence for unsuitable cases.

5.2 Two-class test cases

We designed some test cases to illustrate each calibration map's characteristics empirically. These test cases are designed with a known distribution and allow us to directly sample the instances. For ease of understanding, we consider a scenario with a univariate feature ($\mathbf{x} \in \mathbb{R}$), so that we can inspect the fitted functions and reliability diagrams by means of x-y plots.

Figure 11 depicts the probability density functions for the synthetic two-class dataset, as well as the posterior class probabilities when predicting any new instances. The generative model is equally likely to produce a positive and a negative, and the distribution of the feature values are mixtures of two normal distributions within each class. Denoting the mixture components as $H = 1$ and $H = 2$, our generative model can be formally written down as:

$$
\begin{aligned}
Y &\sim \text{Bernoulli}(0.5) \\
H &\sim \text{Bernoulli}(0.5) \\
X \mid Y = 1, H = 1 &\sim \text{Gaussian}(-4, 960) \\
X \mid Y = 1, H = 2 &\sim \text{Gaussian}(64, 1280) \\
X \mid Y = 2, H = 1 &\sim \text{Gaussian}(4, 980) \\
X \mid Y = 2, H = 2 &\sim \text{Gaussian}(-72, 1024)
\end{aligned}
$$

This generative assumption allows us to approximate the 'true' reliability diagram closely for every test case and provides better insights into each calibration map. Specifically, we generate a large number of random samples and use small bin sizes to approximate a smooth reliability diagram.

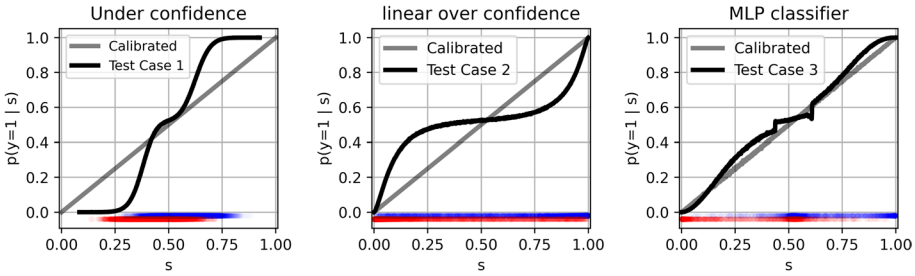


Fig. 12 The reliability diagrams of the three uncalibrated classifiers. The left figure shows the $\hat{\mathrm{f}}_{\mathrm{under}}$ case, where the classifier predicts a probability closer to 0.5 compared with the Bayes-optimal case (Fig. 11, right), resulting in a reliability diagram with steeper slope. The middle figure shows the $\hat{\mathrm{f}}_{\mathrm{over}}$ case, where the predicted probabilities are pushed towards 0 and 1 respectively, resulting in a shallower reliability curve. The right figure is obtained with a MLP classifier ($\hat{\mathrm{f}}_{\mathrm{MLP}}$), where the reliability diagram shows over-confident and under-confident predictions around different regions

We consider three cases for binary classification, each of which may occur in practice. The first two cases are hypothetical over/under-confident scenarios using a logistic regression model of the form $\hat{\mathrm{f}}(x) = (1 + \exp(-a \cdot x))^{-1}$. Such a model puts the decision boundary at $x = 0$, which is nearly Bayes-optimal for the synthetic data. The model confidence can be moderated with the parameter a, and we will use the following two cases:

$$\hat{\mathrm{f}}_{\mathrm{under}}(x) = \frac{1}{1 + \exp(-0.01 \cdot x)}$$
$$\hat{\mathrm{f}}_{\mathrm{over}}(x) = \frac{1}{1 + \exp(-0.05 \cdot x)}$$

The third case is obtained with an MLP classifier as implemented by the scikit-learn Python package. The model is specified with a two layer structure with 128 hidden units per layer. The activation is selected as the ReLU function. The remaining settings are left at their default values. 10, 000 instances are sampled to train the model.

The smoothed reliability diagrams for these three cases are shown in Fig. 12. In the under-confident case the output probabilities barely reach the outside of the interval [0.1, 0.9]. The reliability diagram shows a higher slope compared with the perfectly calibrated one (the diagonal). In contrast, the over-confident case has a shallower slope around the middle compared with the diagonal. It further causes the reliability diagram to have sharp transitions while approaching the extremes. For the MLP case, we can observe both under-confident and over-confident regions throughout the reliability diagram. This specific MLP classifier tends to be under-confident around $s \in [0, 0.1]$ and $s \in [0.65, 1]$, and mostly over-confident elsewhere.

5.3 Calibration for binary classification

We start with methods that can learn calibration maps for binary problems. In binary classification, probability estimates are scalars, so the calibration maps are functions mapping from [0, 1] to [0, 1]. Here we describe four widely applied methods in detail: empirical binning, isotonic regression, logistic calibration and Beta calibration. Other approaches will be be briefly mentioned at the end of this section.

Fig. 13 Examples of empirical binning calibration with 10 equal-width bins. As the binning approach can only give discrete outputs, we can observe the probabilities after calibration are no longer continuous and are only presented with 10 values or less (the under-confidence case has fewer values as there are no observations in bins closer to the extremes. The reliability diagrams show that the calibration method works well with all three test case, with most points being close to the calibrated diagonal

5.3.1 Empirical binning

Empirical binning (Zadrozny & Elkan, 2001; Naeini et al., 2015) is one of the simplest methods to build a calibration map. It directly connects to the previously introduced ECE measure, which evaluates the calibration level by calculating the empirical frequencies within a set of score intervals. Therefore, binning approaches can directly optimise a variant of ECE depending on how the binning scheme is selected.

For a typical setting, the empirical binning approach comes with the following parameters. (1) Bins: $\{\mathbb{B}_1, \dots, \mathbb{B}_M\}$, where each bin $\mathbb{B}_m \subset [0, 1]$ is an interval of probability range, and $\cup_{m=1}^{M} \mathbb{B}_m = [0, 1]$. (2) Probability estimates in the bins: $\mathbf{a} = \{a_1, \dots, a_M\}$, where $a_m \in [0, 1]$ specifies the calibrated probability value in each bin. We usually assume that these bins do not overlap with each other. To make a prediction based on an uncalibrated probability s, empirical binning uses the following functional form:

$$\mathsf{g}(s; \mathbb{B}_1, \dots, \mathbb{B}_M, \mathbf{a}) = a_m \qquad \text{if } s \in \mathbb{B}_m \tag{3}$$

Here $s \in \mathbb{B}_m$ means that the uncalibrated probability s falls within $\mathbb{B}_m$.

Fitting a binning calibration map can be done by optimising $\mathbf{a}$ using ECE as a loss function:

$$\mathbf{a}^* = \arg\min_{\mathbf{a}} \mathsf{L}(\mathbf{a}) \tag{4}$$

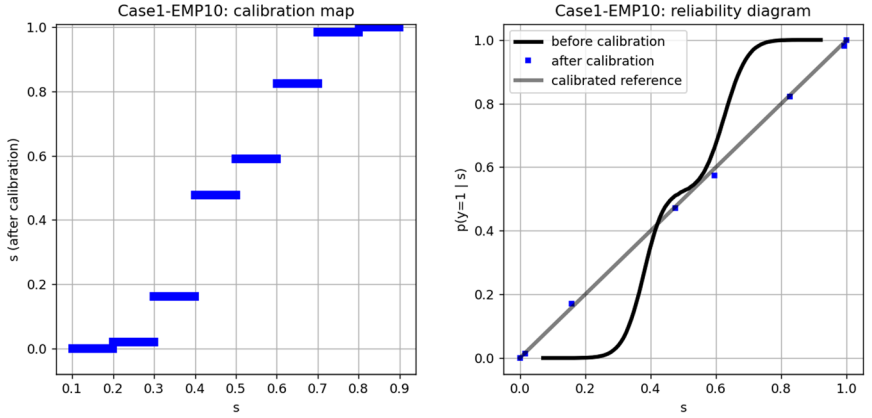
$$\mathsf{L}(\mathbf{a}) = \mathsf{ECE}(\mathbf{a}) = \sum_{m=1}^{M} \left| \bar{\mathsf{y}}(\mathbb{B}_m) - a_m \right| \tag{5}$$

Here $\bar{\mathsf{y}}(\mathbb{B}_m)$ represents the empirical frequency within $\mathbb{B}_m$ as defined in the previous section. The equation hence indicates that we want to assign each a_m according to the empirical frequency of the labels within each bin $\mathbb{B}_m$.

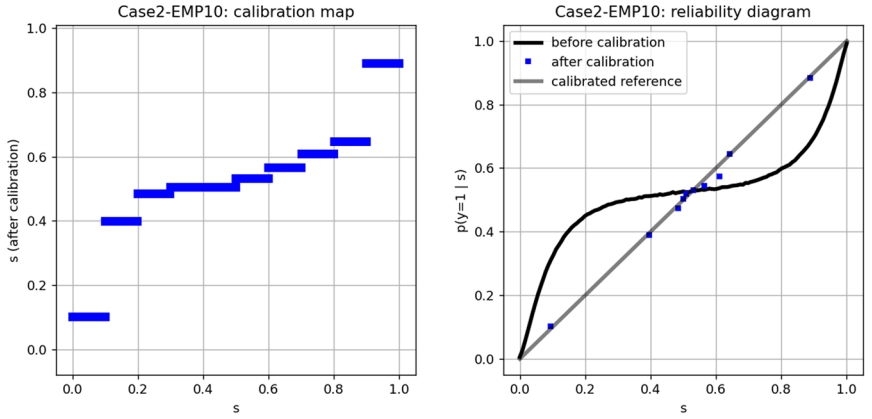
We empirically analyse the binning approach with a simple setting of $M = 10$ equal-width bins. The results with the three test cases are shown in Fig. 13. The binning approach performs well on most points, as the reliability diagram are close to the diagonal after calibration. However, it should be noted that the output probabilities can only take M different values. Therefore, while the final probabilities are calibrated, we can no longer distinguish instances that receive different predictions from the uncalibrated model. In other words, binning approaches can reduce the calibration loss at the cost of a higher grouping loss (Kull & Flach, 2015).

While here we illustrated the binning approach with equal-width bins, it is possible to use other binning schemes such as equal-frequency binning. For example, (Naeini et al., 2015) propose a Bayesian binning approach.

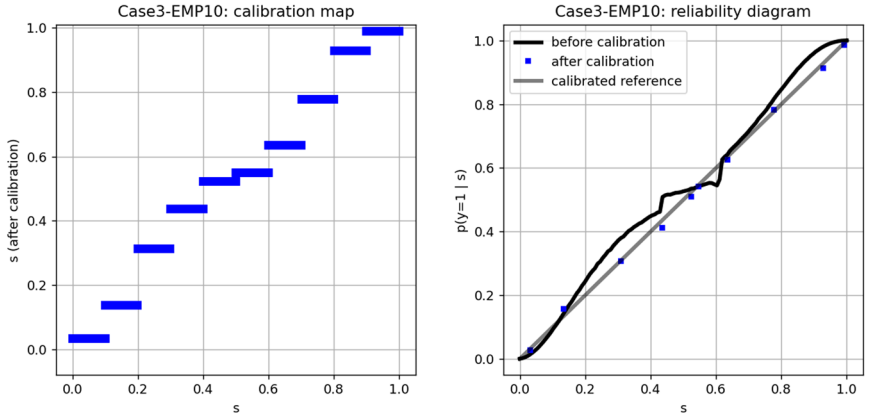
To summarise, the main advantages of binning approaches are as follows: (1) They can be used to directly minimise a specific variant of ECE, hence are useful when calibration loss is of primary concern. (2) Training is reasonably fast and straightforward. (3) Given its non-parametric nature, it does not have any constraint on the form of the calibration



(a) Test case 1: under-confidence



(b) Test case 2: over-confidence



(c) Test case 3: MLP classifier

Fig. 14 Examples of isotonic calibration trained with 10,000 samples. Compared with empirical binning, we can see the isotonic approach can provide a larger variety of outputs in the continuous interval of [0, 1], and different bins can be automatically learnt according to the observations. The calibration method works reasonably well on all three test cases. The over-confidence case still has some errors around around the middle-left and middle-right boundaries, due to the lack of training points within these areas

maps and therefore can capture arbitrary shapes according to the given bins. In addition, other binning strategies can also be applied to deal with different datasets and uncalibrated models.

5.3.2 Isotonic calibration

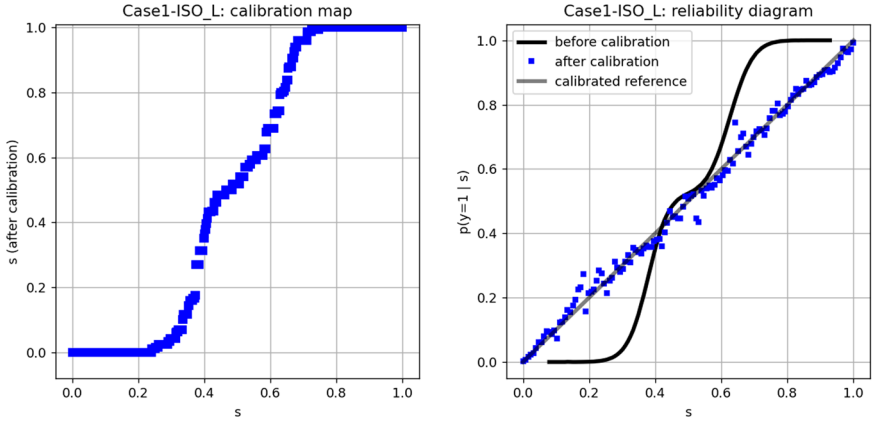
Calibration by means of isotonic regression is another widely used non-parametric approach (Barlow & Brunk, 1972; Niculescu-Mizil & Caruana, 2005; Naeini & Cooper, 2016). The isotonic approach was originally proposed to solve univariate regression tasks where the function were required to be monotonic. In the case of probability calibration, if we assume the uncalibrated model has a monotonic reliability diagram, then isotonic regression is a suitable approach to achieve a better level of calibration.

Isotonic regression achieves a monotonic fit by estimating a set of non-decreasing constant segments, corresponding to a set of bins of varying width. Isotonic regression can therefore be described with the following parameters: (1) Bin boundaries: $\mathbf{b} = (b_1, \ldots, b_M)$, $b_j \in [0, 1], b_j < b_{j+1}$, and (2) Edge values: $\mathbf{v} = (v_1, \ldots, v_M)$, $v_j \in [0, 1], v_j \leq v_{j+1}$. As indicated by the parameters, isotonic regression is quite close to an empirical binning approach. The main difference is that instead of having a pre-defined set of bin edges, isotonic regression learns these bin boundaries from the data. As a result, isotonic regression can have an variable number of bins depending on the training set. To calibrate a probability s, isotonic regression uses the following predictive function:

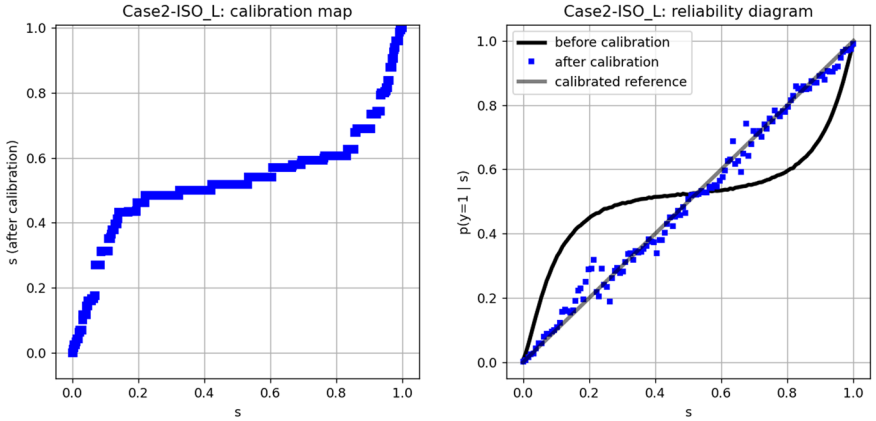
$$\mathbf{g}(s;\mathbf{b},\mathbf{v}) = \sum_{m=1}^{M-1} \mathtt{I}(s \geq b_m) \cdot \mathtt{I}(s < b_{m+1}) \cdot m_j \tag{6}$$

The three test cases on istonic regression are shown in Fig. 14. To compare with the empirical binning approach again, while in the empirical case we have a constant prediction within each bin, we can see the prediction of the isotonic approach depends on the two values on the bin edges. In the case where the two bin edges do not share the same value, isotonic regression performs linear interpolation to calculate the predicted value. This behaviour is due to the way that the isotonic approach is estimated. To ensure the estimated function is monotonically increasing, the learning algorithm starts with the first bin edge as the minimal feature value (0 in the case of an isotonic calibration) and extends the following bin edges until it sees a higher target value (this is also referred to as 'pooling adjacent violators' (Fawcett & Niculescu-Mizil, 2007)). It can be proven that, with monotonic sets of $\mathbf{b}$ and $\mathbf{s}$, the learning algorithm is equivalent to optimising the following loss (Ayer et al., 1955):

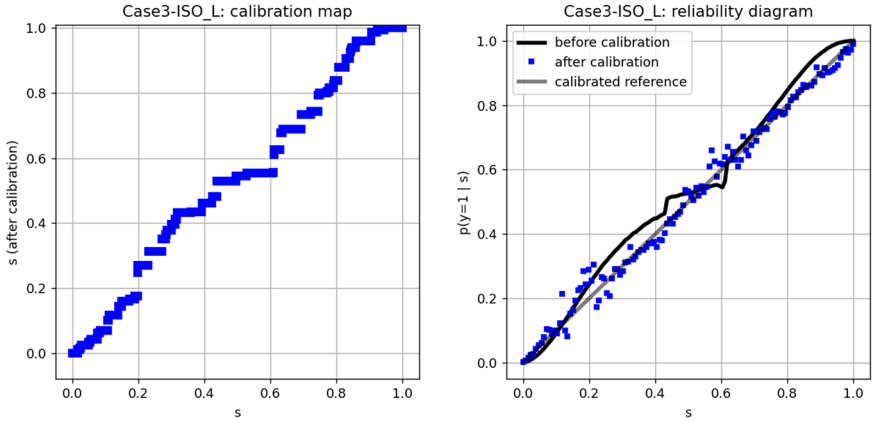
$$\mathsf{L}(\mathbf{b},\mathbf{v}) = \frac{1}{N} \sum_{i=1}^{N} \Big(\mathbf{g}(s_i;\mathbf{b},\mathbf{v}) - y_i \Big)^2 \tag{7}$$



(a) Test case 1: under-confidence



(b) Test case 2: over-confidence



(c) Test case 3: MLP classifier

Fig. 15 Examples of the Platt scaling approach. While it improves the under-confidence case well, it fails to do much with the over-confidence case. This is due to the fact that the sigmoid shape can not push the probabilities towards the middle hence can not fix over-confidence. Such result can also be observed with the MLP test case, where Platt scaling actually makes the level of calibration worse around several regions

Therefore, when used as a probability calibration map, isotonic regression minimises the Brier score on the training set. Instead of applying isotonic regression directly, (Naeini & Cooper, 2016) also consider building an ensemble calibration map by combining a set of near-isotonic functions. Recently, (Allikivi & Kull, 2019) propose a Bayesian approach to improve isotonic regression, with demonstrated improvements on binary tasks.

Isotonic regression is a compelling approach for probability calibration, with the following advantages: (1) When the monotonic assumption is suitable, isotonic regression can find the optimal bin edges for Brier score on the training sets, which is also known to give the convex hull in ROC analysis. (2) The non-parametric nature avoids catastrophic model misfit. On the other hand, isotonic regression has some drawbacks also: (1) By predicting constant values within bins, it inevitably increases grouping loss. (2) When the uncalibrated model has a non-monotonic reliability diagram, isotonic regression can only give sub-optimal results (i.e. large grouping loss). (3) While it performs well on large datasets, isotonic regression's training time and memory consumption are also high on large datasets. (4) Isotonic regression can result in the first and last bins having full confidence (0 and 1), which can be problematic for certain datasets and evaluation metrics (e.g. log-loss). It is therefore common to clip the outputs from isotonic regression with a small ϵ (i.e., 0 to ϵ and 1 to $1-\epsilon$).

5.3.3 Platt scaling

Platt scaling is probably one of the most widely known approaches for probability calibration, partly due to the popularity of SVMs around the year 2000 (Platt, 2000). As SVMs are optimised with hinge loss, they can only output scores on the samples based on the estimated margins. Given that the margins are real scalars by definition, John Platt proposed to transform the scores into probability estimates with logistic regression. Platt scaling hence has the same parameters as a univariate logistic regression model: a shape parameter $w \in \mathbb{R}$, and a location parameter $b \in \mathbb{R}$. The predictive function is then:

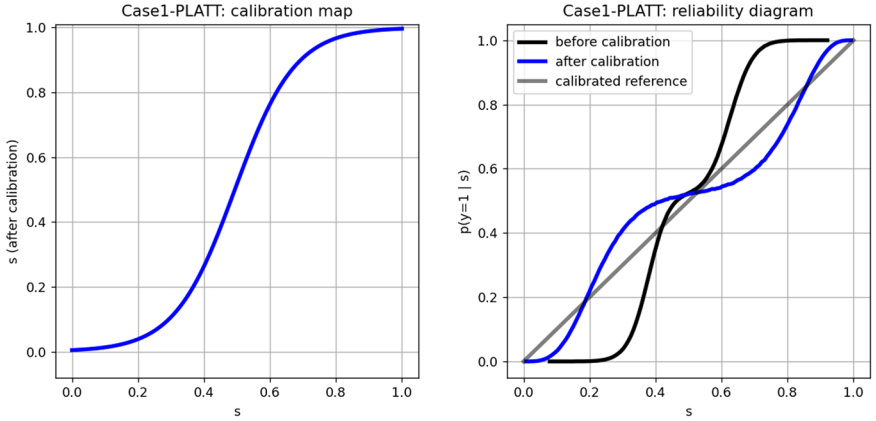
$$\mathrm{g}(s;w,b) = \frac{1}{1+\exp(-w \cdot s - b)} \tag{8}$$

The parameters can be estimated by optimising the log-likelihood on the training set, so Platt scaling can be seen as optimising log-loss in the calibration context:

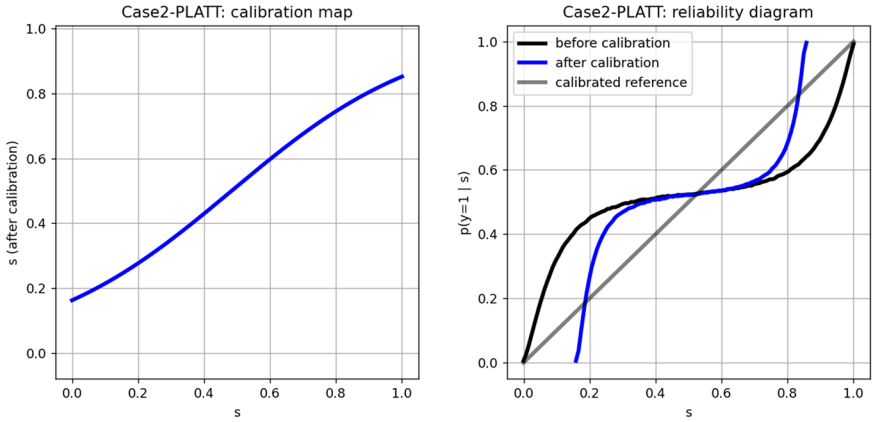
$$\mathrm{L}(w,b) = \frac{1}{N}\sum_{i=1}^{N} \ln\left(\sum_{j=1}^{2} -\mathrm{I}(y_i = j)\cdot \mathrm{g}(s;w,b)\right) \tag{9}$$

Platt also included a technique to prevent overfitting and improve the generalisation of calibration, which is introduced as one of the training schemes at the beginning of this section. The test results are shown in Fig. 15.

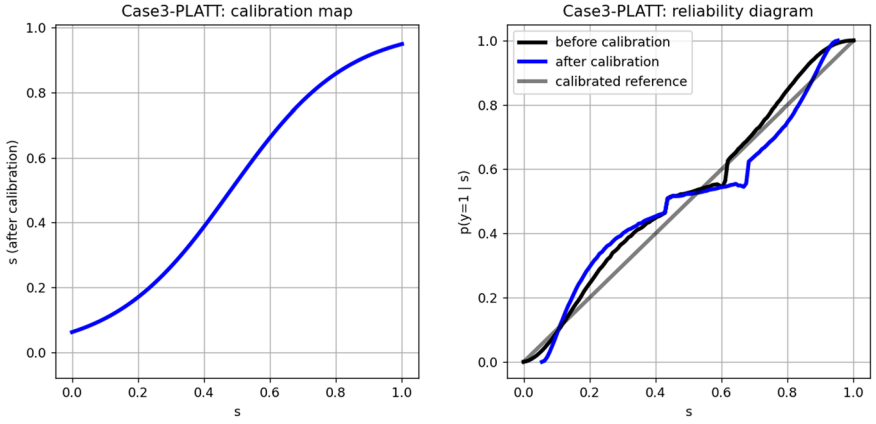
In general, Platt scaling is a practical calibration approach with the following advantages: (1) As it is based on logistic regression, implementation is straightforward. (2) The



(a) Test case 1: under-confidence



(b) Test case 2: over-confidence



(c) Test case 3: MLP classifier

Fig. 16 Examples of Beta calibration. The Beta approach deals with the under-confident case fairly well, as the reliability diagram follows the diagonal closely after calibration. Regarding the over-confident case, Beta calibration manages to put more corrections than the Platt scaling around the [0, 0.5] region, as it pushes the reliability diagram towards the diagonal. However, it doesn't help much in [0.5, 1.0] region. On the MLP case, Beta provides some improvements around the [0.6, 1.0] region, and matches the reliability diagram to the diagonal. Here we can see that the reliability diagrams after calibration are almost identical for the over-confidence and under-confidence test cases. This observation is because the Beta calibration method fits the map on the logarithm of the uncalibrated probabilities, and therefore reverts the sigmoid functions of the pseudo-models in both test cases. The calibration method then finds similar solutions as the pre-sigmoid parts are both linear models on the feature. This result suggests that the Beta calibration method can effectively fix the over/under-confidence caused by a sub-optimal linear-sigmoid operation

standard logistic regression has a convex loss. The overall training process can, therefore be quite fast. On the negative side, Platt scaling has two main drawbacks: (1) The calibration map is restricted to a sigmoid shape, which effectively pushes probabilities away from the centre and may lead to over-confidence. (2) By definition, the logistic regression model considers the input space to be a real scalar space. While such an assumption is suitable for the margins obtained from an SVM, it is less appropriate for probabilistic classifiers, as the input is bounded by the interval [0, 1]. Hence, certain transformations are required when applying Platt scaling to calibrate probabilistic classifiers, such as the logit transform.

5.3.4 Beta calibration

Beta calibration is specifically designed for probabilistic classifiers and alleviates both drawbacks of Platt scaling (Kull et al., 2017a, b). While the latter can be derived from first principles by assuming that within each class the scores are normally distributed with the same variance, Beta calibration instead assumes two Beta distributions. This gives a richer family of calibration functions with three parameters: two shape parameters $a, b \in \mathbb{R}$, and a location parameter $c \in \mathbb{R}$. The calibration function is then given as:

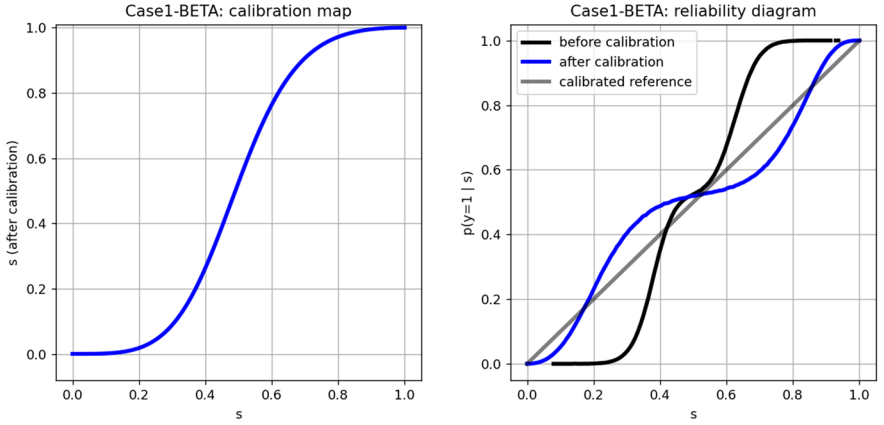
$$\mathsf{g}(s;a,b,c) = \frac{1}{1+\exp\big(-a\cdot\ln s + b\cdot\ln(1-s) - c\big)} \tag{10}$$

In comparison with Platt scaling, Beta calibration is a bivariate logistic regression model, where the two features are $\ln s$ and $\ln(1-s)$ respectively. Similar cross-validation training is also adopted as proposed in Kull et al. (2017a).

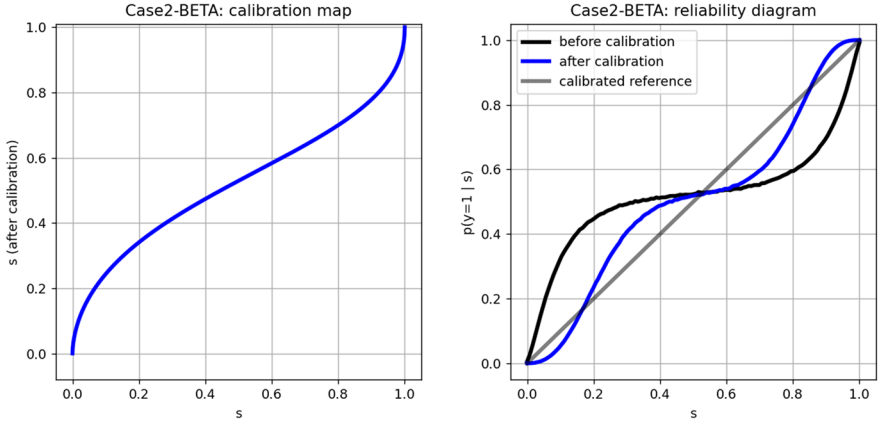
Given its close relationship to logistic regression, Beta calibration also optimises the log-loss and the objective function can be given with the log-likelihood:

$$\mathsf{L}(a,b,c) = \frac{1}{N}\sum_{i=1}^{N}\ln\Big(\sum_{j=1}^{2} -\mathtt{I}(y_i = j)\cdot\mathsf{g}(s;a,b,c)\Big) \tag{11}$$

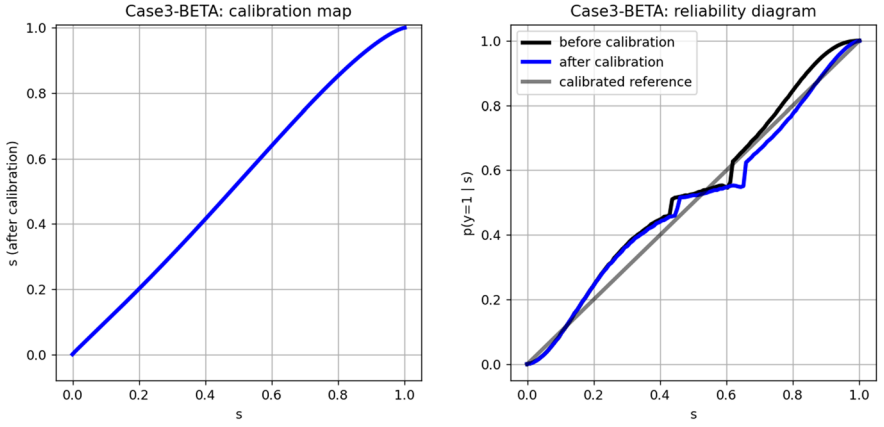
Figure 16 demonstrates Beta calibration on the three test cases. The advantages of Beta calibration include: (1) Compared with Platt scaling, it allows a richer family of calibration maps including inverse sigmoids and the identity map. The latter is particularly useful to prevent over-calibration and apply unnecessary adjustments to already calibrated probabilities. (2) Like with Platt scaling, implementation is straightforward. (3) With the Beta assumption, the calibration map supports any two-class probabilistic classifiers, and the calibration map is learnt from a set of functions that map from [0, 1] to [0, 1]. However,



(a) Test case 1: under-confidence



(b) Test case 2: over-confidence



(c) Test case 3: MLP classifier

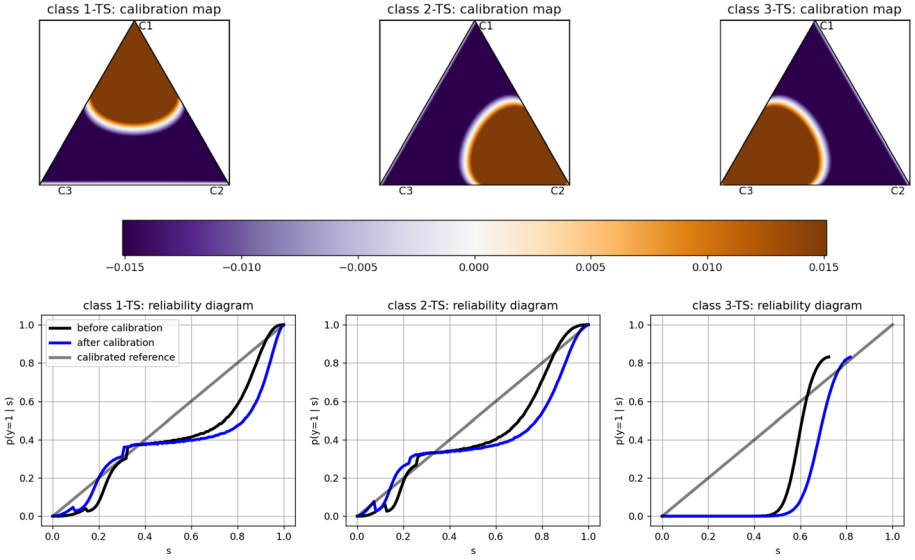


Fig. 17 MLP classifier with Temperature scaling (top: class-wise calibration map, bottom: class-wise reliability diagram). While Temperature scaling manages to improve the reliability diagram around the [0.1, 0.4] region for classes 1 and 2, we can see it actually makes things worse elsewhere. As the calibration maps show, Temperature scaling applies the same calibration map to all three classes despite them having different class-wise reliability diagrams, which is the main reason for the sub-optimal results

Beta calibration's main drawback is that it can only model specific calibration maps, so if the original classifier is severely uncalibrated and requires a complicated calibration map, it might be preferable to choose non-parametric methods instead.

5.4 Calibration for multi-class classification

We now proceed with calibration maps that can work natively with a multi-class classification task, without the need for binarisation. We define a three-class test case for the purpose of examining multi-class calibration with a multi-layer perceptron (MLP). We modify the previous synthetic dataset to add an additional class, such that all classes are equi-probable and the distribution of the feature for the instances of class 3 is Gaussian also:

$$
\begin{aligned}
Y &\sim \text{Categorical}\left(\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]\right) \\
H &\sim \text{Categorical}(0.5) \\
X \mid Y = 1, H = 1 &\sim \text{Gaussian}(-4, 960) \\
X \mid Y = 1, H = 2 &\sim \text{Gaussian}(64, 1280) \\
X \mid Y = 2, H = 1 &\sim \text{Gaussian}(4, 980) \\
X \mid Y = 2, H = 2 &\sim \text{Gaussian}(-72, 1024) \\
X \mid Y = 3 &\sim \text{Gaussian}(0, 8)
\end{aligned}
$$

We visualise the calibration maps in the probability simplex (one for each class). An example can be seen in the top row of Fig. 17, where the heat maps indicate the change of

probability for each class after calibration. For the figure on the left, each point within the simplex represents an uncalibrated probability vector, a positive value (i.e., red colour) on that point indicates that the calibration map increases the probability value of class 1 in the vector and a negative value (i.e., blue colour) shows that the probability value on class 1 is reduced. The white line indicates where the probability values for class 1 remain (almost) unchanged. Class 2 and class 3 calibration maps are also depicted. The bottom row of each figure shows the class-wise reliability diagrams.

5.4.1 Temperature scaling

Temperature scaling has become one of the default approaches to provide better-calibrated probabilities, particularly in deep learning research (Hinton et al., 2015; Guo et al., 2017). Instead of simply applying the softmax operation at the last layer to obtain a probability vector on the classes, Temperature scaling assumes a single temperature parameter $t \in \mathbb{R}$ and applies a linear operation before the softmax. Denoting $\mathbf{z} = [z_1, \ldots, z_K] \in \mathbb{R}^K$ as the K-dimensional real vector before softmax, the probability output after Temperature scaling is given as:

$$\mathsf{c}_j(\mathbf{z};t) = \frac{\exp\left(-z_j/t\right)}{\sum_{j=1}^{K} \exp\left(-z_j/t\right)} \tag{12}$$

Therefore, conventional softmax can be seen as the case of scaling with a fixed temperature of $t = 1$.

If we want to apply Temperature scaling to a generic probabilistic model where z is not accessible, we can also use the logit transform on the uncalibrated probability vector to obtain a real vector:

$$\begin{aligned} z_j &= \mathsf{logit}_j(\mathbf{s}) \\ &= \ln \frac{s_j}{s_K} \end{aligned}$$

Here, K is selected as the reference class to compute the probability ratio with other classes. The parameter is also normally estimated with the log-loss, and the objective function is given as:

$$\mathsf{L}(t) = \frac{1}{N} \sum_{i=1}^{N} \ln\left(\sum_{j=1}^{K} -\mathrm{I}(y_i = j) \cdot \mathsf{g}_j(\mathbf{z}_i;t) \right) \tag{13}$$

Figure 17 gives the results on the MLP test case. The most distinctive characteristic of Temperature scaling is that it only has a single parameter, which can be seen as both an advantage and a disadvantage. On the positive side, a single parameter restricts the space of calibration maps and can prevent overfitting for small datasets with many classes. On the negative side, Temperature scaling can be sub-optimal when the function space of the calibration maps does not include the right reliability diagram.

5.4.2 Vector scaling

Where Temperature scaling only has a single parameter across all classes, it is natural to extend the parameters to support a richer form of calibration maps. Vector scaling (Guo

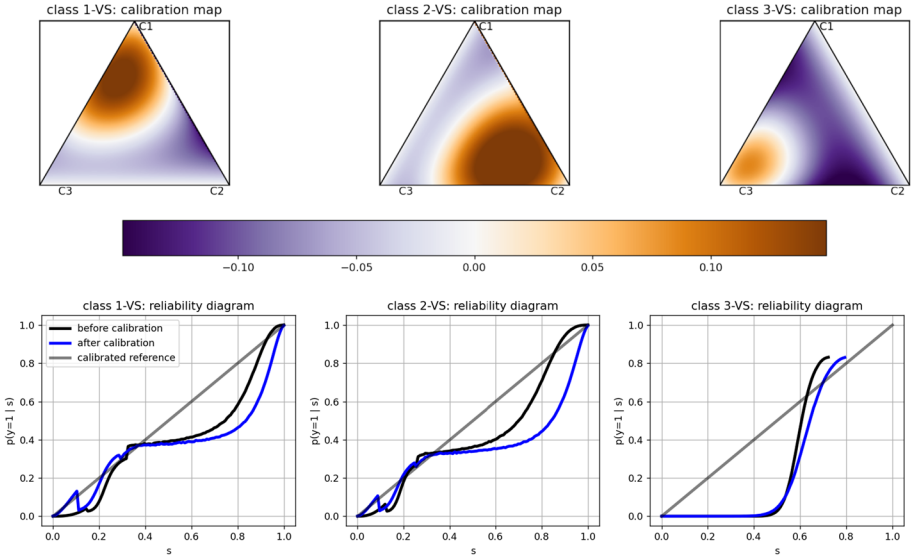


Fig. 18 Class-wise calibration maps and reliability diagrams for the MLP classifier with Vector scaling. The most obvious change from Temperature scaling is that now the calibration maps are different for each class. However, the calibration maps still have a circular shape as Vector scaling only adjusts one dimension of the logit vector for each class. As a result, the reliability diagram is close to the ones of Temperature scaling. Some minor improvements can be seen with class 3, where Vector scaling does not make it quite as bad as for Temperature scaling

et al., 2017) allows a different temperature for each class, denoted as $\mathbf{w} \in \mathbb{R}^K$, as well as adding a set of intercept parameters $\mathbf{b} \in \mathbb{R}^K$ for the K classes. The predictive function is given as ($\mathbf{z}$ can also be obtained with the logit transform for generic probability classifier, as in Temperature scaling):

$$\mathsf{c}_j(\mathbf{z};\mathbf{w}) = \frac{\exp\left(-w_j \cdot z_j - b_j\right)}{\sum_{j=1}^{K} \exp\left(-w_j \cdot z_j - b_j\right)} \tag{14}$$

And the objective function to optimise log-likelihood is similar to the one for Temperature scaling, but with more parameters:

$$\mathsf{L}(\mathbf{w},\mathbf{b}) = \frac{1}{N}\sum_{i=1}^{N} \ln\left(\sum_{j=1}^{K} -\mathrm{I}(y_i = j) \cdot \mathbf{g}_j(\mathbf{z}_i;\mathbf{w},\mathbf{b})\right) \tag{15}$$

The test result is given in Fig. 18. Vector scaling can be seen as a middle point between the quite restrictive Temperature scaling and the general approach of Matrix scaling discussed below. Hence it balances the risk of overfitting and the richness of available calibration maps.

5.4.3 Matrix scaling

Matrix scaling (Guo et al., 2017) removes all the parameter constraints on Temperature scaling and Vector scaling. Similar to a fully connected layer, it has a matrix

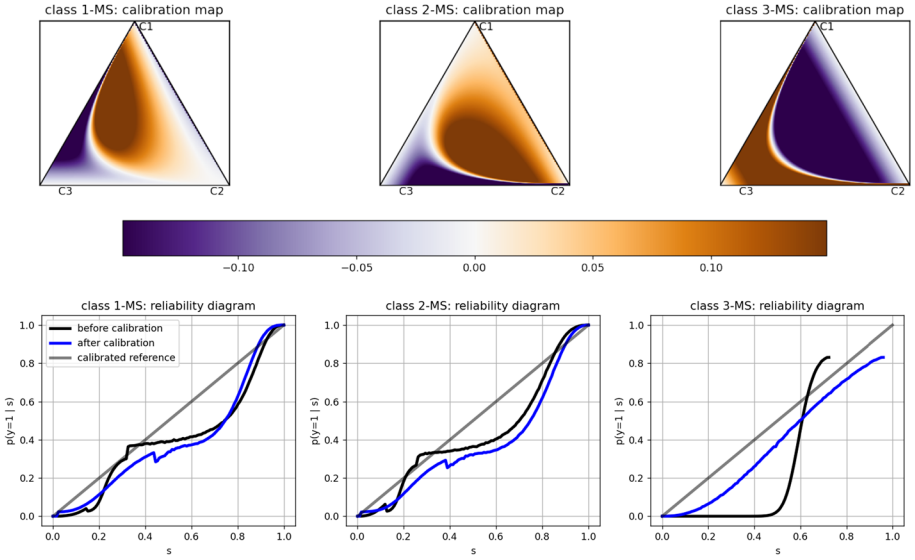


Fig. 19 Class-wise calibration maps and reliability diagrams for the MLP classifier with Matrix scaling. As Matrix scaling is capable of providing fully linear adjustments on the logits for each class, we can see the class-wise calibration map no longer has the circular shape. While there are still minor improvements on the reliability diagram of class 1 and class 2, class 3 has gained a significant improvement

parameter $(\mathbf{w}_1, \cdots, \mathbf{w}_K), \mathbf{w}_j \in \mathbb{R}^K$ together with the intercept parameter $\mathbf{b} \in \mathbb{R}^K$. The predictive function is ($\mathbf{z}$ can also be obtained with the logit transform, as in Temperature scaling):

$$\mathsf{c}_j(\mathbf{z};\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{b}) = \frac{\exp\left(-\mathbf{w}_j^T\mathbf{z} - b_j\right)}{\sum_{j=1}^K \exp\left(-\mathbf{w}_j^T\mathbf{z} - b_j\right)} \tag{16}$$

and the objective function is:

$$\mathsf{L}(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{b}) = \frac{1}{N}\sum_{i=1}^{N} \ln\left(\sum_{j=1}^{K} -\mathrm{I}(y_i = j) \cdot \mathsf{g}_j(\mathbf{z}_i;\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{b})\right) \tag{17}$$

As indicated in Fig. 19, Matrix scaling can support a broader family of calibration maps than Vector scaling and Temperature scaling, but may be vulnerable to overfitting as it can be seen as another fully connected layer.

5.4.4 Dirichlet calibration

Dirichlet calibration is the multi-class extension of Beta calibration (Kull et al., 2019). Assuming Dirichlet distributions for the predicted probability vectors within each class, Dirichlet calibration is determined by a set of coefficient vectors $(\mathbf{w}_1, \cdots, \mathbf{w}_K), \mathbf{w}_j \in \mathbb{R}^K$, and a set of intercepts $\mathbf{b} \in \mathbb{R}^K$. The calibrated probability on the j-th class is obtained as:

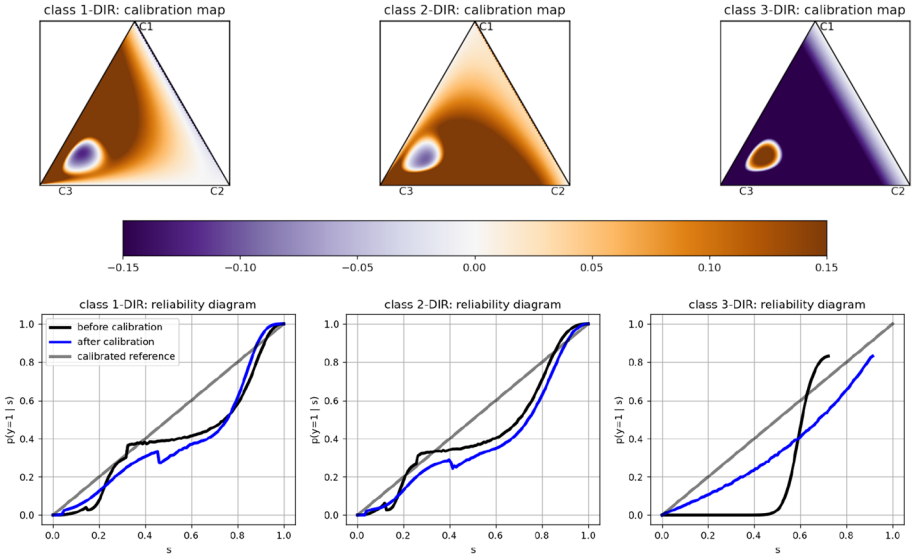


Fig. 20 Class-wise calibration maps and reliability diagrams for the MLP classifier with Dirichlet calibration. The overall results on the class-wise reliability diagrams are close to those obtained with Matrix scaling. The major differences are seen in the calibration maps, where we can clearly observe a local region for the changes on class 3. Importantly, this region for the Dirichlet calibration approach doesn't cover the area where the predicted probability of class 3 is close to 1. This is more reasonable compared with Matrix scaling, as both the Bayes-optimal predictions and the MLP predictions for class 3 are capped around 0.8

$$\mathrm{g}_j(\mathbf{s};\mathbf{w}_1,\ldots,\mathbf{w}_K,\mathbf{b}) = \frac{\exp(-b_j - \mathbf{w}_j^T \ln \mathbf{s})}{\sum_{j=1}^{K} \exp(-b_j - \mathbf{w}_j^T \ln \mathbf{s})} \tag{18}$$

This function can be seen as a K-dimensional logistic regression on the log-probabilities (i.e., the sufficient statistics of the Dirichlet distribution). Therefore, to apply Dirichlet calibration to a deep neural network, one can first obtain a probability vector using the softmax function, then apply Dirichlet calibration as with other probabilistic classifiers. The objective function is:

$$\mathrm{L}(\mathbf{w}_1,\ldots,\mathbf{w}_K,\mathbf{b}) = \frac{1}{N}\sum_{i=1}^{N} \ln\left(\sum_{j=1}^{K} -\mathrm{I}(y_i = j)\cdot \mathrm{g}_j(\mathbf{s}_i;\mathbf{w}_1,\ldots,\mathbf{w}_K,\mathbf{b})\right) \tag{19}$$

The test results are given in Fig. 20. Similar to Beta calibration, one of the advantages of Dirichlet calibration is that it also supports the identity map, so it can also prevent over-adjusting calibrated results. As the number of parameters is similar to that of Matrix scaling, the Dirichlet calibration can also overfit on small datasets. The authors propose the ODIR (Off-Diagonal and Intercept Regularisation) approach to address this issue further and report various performance improvements in their experiments.

5.5 Extended methods for post-hoc classifier calibration

In this section we discuss more recent work extending the standard calibration methods discussed above.

Leathart et al. (2017) aim to build calibration maps locally instead of a global calibration map and proposes using modified logistic model trees to build an ensemble calibration method. A decision tree is learnt to separate the original feature space into multiple regions, where a local calibrator can then be estimated using the corresponding outputs and labels. Kumar et al. (2019) propose a combined approach of scaling and binning methods to achieve better calibration with smaller calibration sets. Instead of calculating the label frequency in binning methods, the authors show that using the average value of a pre-fit continuous calibration map gives better calibration. The authors also indicate that a debiased estimator can provide better calibration error in sample complexity. Wang et al. (2019) discuss how to build flexible univariate non-decreasing calibration maps from shape-restricted polynomial regression models. The resulting method provides a useful alternative when methods like Plat scaling and Beta calibration cannot approximate the actual reliability diagram. Wenger et al. (2020) investigate non-parametric post-hoc approaches that can support richer forms of calibration map for a multi-class setting. The idea is to employ the framework of Gaussian processes and model the calibration map as latent distribution of functions. The family of calibration maps can thus be richer with the selection of suitable kernels. Zhang et al. (2020) propose an approach to build an ensemble of base calibrators, which can improve the capacity of the calibration maps while preserving the accuracy. The authors also adapt kernel density estimation as an alternative approach to evaluating calibration error. Ma and Blaschko (2021) consider the confidence calibration setting and propose to improve a calibration map with a ranking model. The authors demonstrate that the ranking model can be used to maintain the performance of the uncalibrated classifier in terms of precision and false positive rate. Patel et al. (2021) tackle the issue of applying class-wise binning calibrators to a multi-class setting. An iterative update method is provided to adjust the bin edges to better calibration, and a sample-sharing approach among similar classes is also introduced for better sample efficiency.

5.6 Methods that can improve calibration during training time

While post-hoc calibration can be applied to most existing classifiers, it is of practical interest to train well-calibrated models in the first place. Here we briefly mention some training-time calibration methods.

Kumar et al. (2018) consider a regularisation approach with kernel mean embedding. The idea is to add a differentiable regularisation term concerning the goodness of calibration, where the kernel mean of the scores is calculated as an approximation of the calibration error. The overall objective function allows the model to achieve better calibration during the training phase. Thulasidasan et al. (2019) show that the Mixup strategy can improve the calibration level when introduced during the training phase. Vicinal points are created by combining existing data points and labels, and then added to the training set. The authors suggest that the improvement in calibration is due to the augmentation effect and label smoothing effect, both from the mixup process and act as a form of regularisation. Kristiadi et al. (2020) consider a Bayesian approach where a Gaussian approximation distribution is fitted to the last layer's weights of a ReLu network. The authors

demonstrated that the method could reduce the over-confidence problem in a classification setting, particularly on data points not commonly seen in the original training set. Wen et al. (2020) propose an efficient ensemble approach that can train multiple deep networks with lower computational and memory costs. The authors show that the obtained ensemble achieves better calibration on out-of-distribution samples at test time.

Other than deep networks, some recent work achieves better calibration on Bayesian nonparametric methods. Milios et al. (2018) propose an alternative approach to train Gaussian Processes for classification tasks. While the traditional GP classifier relies on sigmoid or probit link functions and requires suitable approximated inference schemes to ensure good performance, the authors propose a modified Dirichlet link function to ensure the level of calibration as well as scalability. Tran et al. (2019) demonstrate that convolution Gaussian process classifiers can be miscalibrated, and propose a solution to mitigate the issue.

5.7 Methods for probability calibration for other predictive settings

The concept of calibration can be naturally generalised to predictive settings other than classification. The most straightforward extension is the calibration of regression outputs. As in the classification scenario, there are multiple definitions of calibration for regression models. The standard definition of calibration comes from the field of quantile regression Koenker and Hallock (2001), where being calibrated means that the predicted quantile matches the marginal distribution of the target variable. As in the case of classification, there are methods designed for training calibrated quantile regression as well as post-hoc approaches that calibrate pre-trained models. Fasiolo et al. (2020) provide an approach to train generalised additive regression models with calibrated quantiles. Cui et al. (2020) propose an approach to train deep regression models with better-calibrated quantiles, where the model parameter is first fit with log-loss and then updated with a loss based on the maximal mean discrepancy Gretton et al. (2012). For the post-hoc setting, Kuleshov et al. (2018) point out that deep regression models' quantiles can be calibrated with binary calibrators such as isotonic regression and yield better performance on downstream tasks such as reinforcement learning. Chung et al. (2021) introduce some modified loss functions to train quantile calibrators with specific emphases, such as balancing between calibration and sharpness.

Regarding the definition of calibration, Song et al. (2019) propose a stronger notion of calibration in regression, where the local conditional distribution is required to be calibrated instead of the global quantile. The authors show that being distribution-calibrated in this sense implies being quantile-calibrated. A post-hoc approach based on the Gaussian process is also introduced, where the GP models a distribution of Beta calibrators conditioning on the original regression outputs. Kuleshov and Deshpande (2022) consider achieving post-hoc distribution calibration by learning a lower dimensional representation of the (uncalibrated) local distributions, and the post-hoc calibrators can then be built with density estimators on these representations and the target values, which avoids the costs of distributional kernels and sparse variational inference. Sahoo et al. (2021) propose the definition of threshold calibration for regression models, which is a stronger notion of calibration than being quantile-calibrated and weaker than the distribution-calibrated definition. Instead of asking for calibration on a marginal quantile or local distribution level, the proposed definition requires the model to be calibrated on a group of distributions that satisfy

a given threshold condition, which ensures optimal decision-making when a threshold is put on the regression outputs.

Calibration of structured outputs, such as sequential outputs, is also of interest for application domains such as natural language processing and time-series modelling. Kuleshov and Liang (2015) propose approaches to simplify the structured output to a two-class problem and employ a well-calibrated binary classifier to improve the calibration level. On the information-retrieval side, the traditional precision-recall curve is not sufficiently similar to a ROC curve to allow a similar calibration property. In contrast, the precision-recall-gain curve proposed by Flach and Kull (2015) can be seen as a proper generalisation of ROC curve obtained by replacing weighted accuracy with F-score.

Conformal prediction (Vovk et al., 2005; Romano et al., 2019; Angelopoulos & Bates, 2021) is another area closely related to the topic of probability calibration. For classification, conformal prediction aims to predict a set of labels that has a probability of $1 - \alpha$ that includes the true label, where α is a user-defined significance level. Therefore, we can see the conformal prediction for classifiers as an alternative definition of calibration to label probability calibration. That is, instead of requiring the probability to be accurate on the label distributions, conformal prediction wants to ensure the calibration on a variable set of labels that are exactly calibrated on the given significance level. One of the approaches to construct a conformal classifier is to use a Venn predictor (Vovk et al., 2003; Vovk & Petej, 2012; Johansson et al., 2018). The framework makes use of the Venn taxonomy to define subsets of the original label space, and requires a classifier to make predictions about each element in the subset. The final conformal prediction can then be calculated using these predictions.

5.8 Methods designed for other research fields and applications

The notion of calibration can also be generalised to other areas where uncertainty quantification is of interest. rRecent work has started to explore other topics in statistics, machine learning and artificial intelligence related to calibration. Here we briefly introduce some of them. A comprehensive analysis of these works is left as future work. Pleiss et al. (2017) draw a link between the level of calibration and the evaluation of classifier fairness. Cobb et al. (2018) propose an approach to modify the lower bound of approximate Bayesian inference so that the final objective function can be calibrated according to a given decision loss. Liu et al. (2018) adopt calibrated deep models to improve zero-shot learning. Menon and Williamson (2018) consider the calibration problem for an anomaly detection setup. Ghandeharioun et al. (2019) consider uncertainty and calibration as a tool to improve explainability for computer vision tasks. And Liu et al. (2019) adopt a calibration procedure to achieve better posterior distributions for a Bayesian non-parametric ensemble model. Ding et al. (2021) consider the problem of improving calibration for semantic segmentation. Guillory et al. (2021); Jiang et al. (2021) show that calibrated classifier can be used to estimate the model performance on OOD test sets.

6 Hypothesis tests for calibration

Even if a classifier is perfectly calibrated, it almost always slightly deviates from calibration on any finite dataset due to sampling effects. Here we consider a method of testing whether the dataset provides sufficient evidence that the classifier is uncalibrated.

Table 8 Instances grouped into 5 bins using equal-frequency binning on the complement of the probabilities predicted for class 1. Column $\mathsf{s}(\mathbb{B}_m)$ shows the probability intervals corresponding to each bin. The probabilities were obtained by considering the multiclass toy problem in Table 3

$\mathbb{B}_m$	$\mathsf{s}(\mathbb{B}_m)$	O_{m1}	E_{m1}	O_{m2}	E_{m2}	O_{m3}	E_{m3}
$\mathbb{B}_1$	[0.0, 0.2]	3.0	5.9	1.0	0.7	3.0	0.4
$\mathbb{B}_2$	(0.2, 0.56]	2.0	3.1	2.0	0.6	1.0	1.3
$\mathbb{B}_3$	(0.56, 0.7]	3.0	2.47	3.0	1.77	1.0	2.77
$\mathbb{B}_4$	(0.7, 0.9]	2.0	1.1	3.0	3.1	2.0	2.8
$\mathbb{B}_5$	(0.9, 1.0]	0.0	0.0	1.0	1.2	3.0	2.8

A common way to test for calibration is to apply goodness-of-fit tests that were originally proposed for logistic regression models, but can be extended to any model that predicts probabilities. For example, one could apply the Pearson chi-squared goodness-of-fit test, which works by grouping instances sharing the same value of the independent variable X and then applying a chi-squared test under the null hypothesis that the observed number of positives in the target variable is equal to the predicted number of positives for each unique value of X.

The issue with this approach is that the higher the number of unique values of X or the higher the number of combinations of values of multiple independent variables, the lower the probability that each group of instances will be well-represented enough for the calculation of the Pearson chi-squared statistic to be reliable. A possible solution to this problem is to group instances into M bins based on percentiles of the predicted probabilities for the positive class (equal-frequency binning) and then apply the Hosmer-Lemeshow (HL) test, which involves calculating the statistic H given by Equation (20) (Hosmer & Lemeshow, 2003).

$$H = \sum_{m=1}^{M} \left(\frac{\left(O_{m+} - E_{m+}\right)^2}{E_{m+}} + \frac{\left(O_{m-} - E_{m-}\right)^2}{E_{m-}} \right), \tag{20}$$

where O_{m+} and O_{m-} are the observed counts of positives and negatives in bin $\mathbb{B}_m$ and E_{m+} and E_{m-} are the expected counts of positives and negatives in bin $\mathbb{B}_m$, calculated as the sums of positive and negative probabilities in the bin, respectively.

The HL test was later extended to evaluate the goodness-of-fit of multinomial logistic regression models, by binning instances based on the percentiles of $\sum_{j=2}^{K} s_{ij} = 1 - s_{i1}$, i.e. the complement of the probabilities assigned to the reference class (Fagerland et al., 2008). Then, similarly to the binary case, once the instances are binned, we calculate O_{mj} and E_{mj}, respectively the observed and expected counts for each class j and bin m. Finally we calculate the C statistic, given by Equation (21), which is expected to follow a chi-squared distribution with $(M-2)(K-1)$ degrees of freedom.

$$C = \sum_{m=1}^{M} \sum_{j=1}^{K} \left(\frac{\left(O_{mj} - E_{mj}\right)^2}{E_{mj}} + \frac{\left(O_{mj} - E_{mj}\right)^2}{E_{mj}} \right). \tag{21}$$

Table 8 shows $M = 5$ bins based on the toy example of Table 3. By applying Equation (21) to these values, we get $C = 25.3$. Finally, to complete the HL test under the null hypothesis that the observed and expected proportions are the same, we compare C to a chi-squared

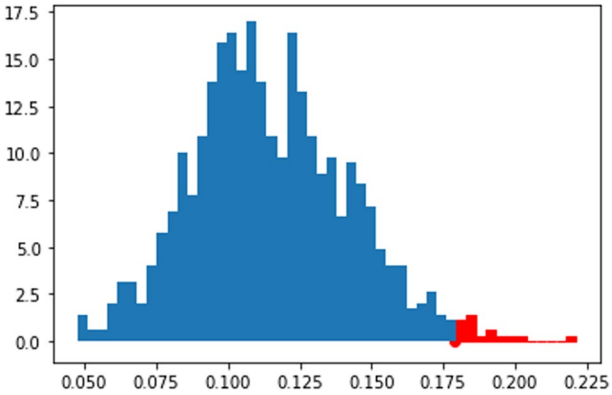


Fig. 21 Distribution of $\text{ECE}_{\text{classwise}}$ values $\phi_1, \ldots, \phi_L$ calculated for every resampled label set $\dot{\mathbf{Y}}_l$ with $l \in \{1, \ldots, L\}$, following the procedure described in Algorithm 1. The red bars represent the region where $\phi_l > 0.1787$

distribution with $(M-2)(K-1) = 6$ degrees of freedom, obtaining a p-value of 0.0003, rejecting the null hypothesis with $\alpha = 0.05$.

This example is of course simply illustrative, as the HL test is not recommended for such small samples, as experiments have shown that it has satisfactory power for samples of over 400 instances (Fagerland et al., 2008). There are several other goodness-of-fit tests, most of them using some sort of binning to calculate a statistic that is assumed to follow a chi-squared distribution. Additionally, as the multiclass HL test, they usually test for classwise calibration. Testing for multiclass calibration is a more complex task and has recently been investigated by Widmann et al. (2019), by using kernel-based measures. These measures were later adapted for regression calibration (Widmann et al., 2022). For details on how these measures can be interpreted as test statistics, we refer the reader to their original papers.

Another approach, proposed by Vaicenavicius et al. (2019), is to test the classifier's probabilities according to an arbitrary measure of calibration. Given a classifier f, we can check if its predictions for a test set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ are calibrated according to a chosen calibration measure $\phi(\mathsf{f}(\mathbf{X}_{\text{test}}), \dot{\mathbf{Y}}_{\text{test}})$, such as ECE, log-loss or Brier score. We discuss a simple resampling-based hypothesis test under the null hypothesis that the classifier's outputs are calibrated (Vaicenavicius et al., 2019). Algorithm 1 details the steps of the hypothesis test procedure.

Algorithm 1 Hypothesis test for calibration

Require: Test predictions $\mathsf{f}(\mathbf{X}_{\text{test}})$; test set labels $\dot{\mathbf{Y}}_{\text{test}}$, Calibration measure ϕ; Number of resampled label sets L; Significance level α;

1: **for** l=1:L **do**
2: Sample y_l, $l \in \{1, \ldots, L\}$, such that each $y_{l,i} \sim \mathsf{Cat}(\mathsf{f}(\hat{\mathbf{x}}_i))$;
3: $\phi_s \leftarrow \phi\left(\mathsf{f}(\mathbf{X}_{\text{test}}), \dot{\mathbf{Y}}_l\right)$;
4: **end for**
5: p-value $\leftarrow$ proportion of ϕ_l among $\phi_1, \ldots, \phi_L$ with $\phi_l > \phi\left(\mathsf{f}(\mathbf{X}_{\text{test}}), \dot{\mathbf{Y}}_{\text{test}}\right)$;
6: **if** p-value $< \alpha$ **then**
7: Reject the null hypothesis and conclude that the classifier is not calibrated;
8: **else**
9: Accept the null hypothesis, i.e. there is not enough evidence that the classifier is not calibrated.
10: **end if**

Let us return to our toy example (Table 3). If we follow Algorithm 1 and choose $\mathsf{ECE}_{\text{classwise}}$ as the calibration measure and $L = 1,000$, we obtain a distribution such as the one in Fig. 21.

Recall that in our example $\mathsf{ECE}_{\text{classwise}} = 0.1787$. The p-value for this test is then $\mathsf{P}\Big(\phi_l > 0.1787\Big) \approx 0.016$, which means we can reject the null hypothesis that the classifier is calibrated according to $\mathsf{ECE}_{\text{classwise}}$. Note that we keep the scores fixed and only resample the labels to perform this test. This means that instances will stay in the same bins and small bins will have low weights for all L resampled $\mathsf{ECE}_{\text{classwise}}$ values.

7 Concluding remarks

In this survey we have given a detailed overview of the principles and practice of classifier calibration. We discussed the motivation, definition, evaluation and related approaches for the calibration of probabilistic classifiers, and touched upon topics beyond the standard classification setting. With the recent proliferation of predictive machine learning, it is becoming more and more important that models are carefully evaluated and if possibly improved on multiple criteria to ensure the overall system robustness. Classifier calibration relates to the criterion of uncertainty quantification, investigating the statistical relationships between model predictions and observed target variables. There are other levels of uncertainty in the overall modelling procedure, including uncertainty pertaining to model parameters when the training set is limited, uncertainty brought about by model mismatch, or uncertainty regarding whether the model should abstain when there might be novel target values that were not seen in the training set – see (Hüllermeier & Waegeman, 2021) for a recent survey on uncertainty in machine learning.

As we hopefully have made clear in this survey, calibration research has a rich history which predates the birth of machine learning as an academic field by decades. It is important to take full advantage of that history, which perhaps isn't always fully acknowledged in recent literature on classifier calibration (e.g., of deep neural networks, which are often over-confident). Recent proposals such as confidence calibration, temperature scaling and expected calibration error have value but are far from the only options and may not be suitable for a particular application. As is often the case in machine learning, the space of options and things to consider is large, and navigating this space requires the right set of concepts and tools, many of which we have discussed in detail in this survey.

It is perhaps useful to point out that, naturally, there are also concepts that are less than helpful when thinking about classifier calibration. One of the most prominent of these is the concept of a decision boundary, which is like describing Mount Everest with a single contour line, halfway up. To faithfully characterise the mountain – which is what is needed for optimal decision making – one needs many contour lines at different elevations. The only case in which the decision boundary is useful is when the class prior (and misclassification costs, if any) won't change after training, in which case only the decision threshold needs calibrating. In all other cases context change is expected and needs to be anticipated with a proper calibration analysis.

If a trained classifier f is going to be used for a task for which it is good enough to be accurate, precise or to perform well according to any contingency matrix-based measure, then there is no particular need to consider calibration in the model's development pipeline. However, if calibrated probabilities are important for any decisions supported by f, the following steps can be taken to implement what was discussed in this paper:

1. If possible, any classifiers considered for a task should be trained using a proper scoring rule as loss function;
    - The resulting models are likely to produce better probabilities than if a non-proper loss function is used, as probability refinement and calibration will be encouraged during training (recall Sect. 3);
    - An alternative is to use a method that encourages calibration during training time, as discussed in Sect. 5.6;
2. With the trained classifiers and, preferably a held-out calibration dataset, the calibration of the probabilities produced by the models can be evaluated using classwise reliability diagrams, to understand the miscalibration patterns, and/or ECE measures (recall Sect. 4);
3. To help define which models are uncalibrated, a hypothesis test for calibration can be applied (recall Sect. 6);
4. If any models produce uncalibrated probabilities, post-hoc calibrators should be used to fix them;
    - The resulting calibrators should also be evaluated using reliability diagrams, ECE measures and hypothesis tests to find out which ones work best for the particular miscalibration patterns found during step 2.

In order to help with the aforementioned steps, we have developed an open-source Python library called PyCalib (Perello-Nieto et al., 2021).[4] This library provides tools to measure the calibration quality of a classifier, multiple visualisation functions to better understand how calibrated is a model, and some calibration methods and pipelines to directly train arbitrary calibrated classifiers.

We close this survey with some open problems. First, although binning is such an important concept in calibration, there is no standard method to decide which type of binning, e.g. equal frequency or equal width, or what number of bins to use. Recent work (Kängsepp et al., 2022) has investigated cross-validation to tune the number of bins used to evaluate a model on a calibration set. Second, a recurrent topic of interest is the fact that post-hoc calibration does not improve robustness against out-of-distribution inputs and what can be done to improve calibration in this context (Ovadia et al., 2019). Finally, and somewhat related to the previous point, even though calibrated probabilities provide estimations of the uncertainty regarding an instance's class given its feature values (aleatoric or first-order uncertainty), classifiers could also be made to predict their uncertainty about the predicted probabilities, which comes from not knowing the true probabilities (epistemic uncertainty). This type of prediction, a distribution of distributions, also called second-order uncertainty (Hüllermeier et al., 2022; Bengs et al., 2022), has been recently treated using inference based on Bayesian methods and credal sets, which are convex sets of probability distributions (Shaker & Hüllermeier, 2021).

Author Contributions TSF: conceptualisation, formal analysis, methodology, software, writing (original draft, and review and editing). HS: conceptualisation, formal analysis, methodology, software, writing (original draft, and review and editing). MPN: formal analysis, methodology, software, visualisation, writing (original draft, and review and editing). RSR: formal analysis, writing (review and editing). MK: conceptualisation, formal analysis, methodology, writing (review and editing). PF: conceptualisation, formal

[4] The documentation of the library can be found at https://classifier-calibration.github.io/PyCalib/.

analysis, methodology, supervision, writing (original draft, and review and editing). (The categories follow the CRediT taxonomy: http://credit.niso.org/)

Funding The work of PF, RSR and MPN was supported by the SPHERE Next Steps Project funded by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant EP/R005273/1]. The work of RSR was funded by the UKRI Turing AI Fellowship [grant EP/V024817/1]. The work of PF and HS was supported by The Alan Turing Institute under EPSRC [grant EP/N510129/1]. The work of PF was partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215. The work of MK was supported by Estonian Research Council [grant PRG1604] and by the Estonian Centre of Excellence in IT (EXCITE), funded by the European Regional Development Fund.

Data availibility The availability of data and material used in this survey can be found at https://github.com/classifier-calibration/additional_material. Further information is available at https://classifier-calibration.github.io/survey/.

Code availability The code to generate measurements, visualisation and results is part of PyCalib. Further information can be found in the documentation of the library https://classifier-calibration.github.io/PyCalib/.

Declarations

Conflict of interest None.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Allikivi, M.-L., & Kull, M. (2019). Non-parametric bayesian isotonic calibration: Fighting over-confidence in binary classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD'19)*, (pp. 68–85).

Angelopoulos, A. N. & Bates, S. (2021). A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint*arXiv:2107.07511.

Ayer, M., Brunk, H. D., Ewing, G. M., Reid, W. T., & Silverman, E. (1955). An empirical distribution function for sampling with incomplete information. *The Annals of Mathematical Statistics, 26*, 641–647.

Barlow, R. E., & Brunk, H. D. (1972). The isotonic regression problem and its dual. *Journal of the American Statistical Association, 67*(337), 140–147.

Bengs, V., Hüllermeier, E., & Waegeman, W. (2022). Pitfalls of epistemic uncertainty quantification through loss minimisation. In A. H. Oh, A. Agarwal, D. Belgrave, & K. Cho (eds.), *Advances in Neural Information Processing Systems*.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review, 78*(1), 1–3.

Bröcker, J., & Smith, L. A. (2007). Increasing the reliability of reliability diagrams. *Weather and Forecasting, 22*(3), 651–661.

Bross, I. (1954). Design for decision. *Science Education, 38*(4), 325.

Chung, Y., Neiswanger, W., Char, I., & Schneider, J. (2021). Beyond pinball loss: Quantile methods for calibrated uncertainty quantification. *Advances in Neural Information Processing Systems, 34*, 10971–10984.

Cobb, A. D., Roberts, S. J., & Gal, Y. (2018). *Loss-Calibrated Approximate Inference in Bayesian Neural Networks*. ICML: Theory of deep learning workshop.

Cui, P., Hu, W., & Zhu, J. (2020). Calibrated reliable regression using maximum mean discrepancy. *Advances in Neural Information Processing Systems, 33*, 17164–75.

DeGroot, M. H., & Fienberg, S. E. (1983). The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society Series D (The Statistician), 32*(1/2), 12–22.

Ding, Z., Han, X., Liu, P., & Niethammer, M. (2021). Local temperature scaling for probability calibration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (pp. 6889–6899).

Fagerland, M. W., Hosmer, D. W., & Bofin, A. M. (2008). Multinomial goodness-of-fit tests for logistic regression models. *Statistics in Medicine, 27*(21), 4238–4253. https://doi.org/10.1002/sim.3202

Fasiolo, M., Wood, S. N., Zaffran, M., Nedellec, R., & Goude, Y. (2020). Fast calibrated additive quantile regression. *Journal of the American Statistical Association*, 1–11.

Fawcett, T., & Niculescu-Mizil, A. (2007). PAV and the ROC convex hull. *Machine Learning, 68*(1), 97–106.

Flach, P. (2012). *Machine learning: The art and science of algorithms that make sense of data*. Cambridge: Cambridge University Press.

Flach, P. (2014). Classification in context: Adapting to changes in class and cost distribution. In *First international workshop on learning over multiple contexts at European conference on machine learning and principles and practice of knowledge discovery in databases (ECML-PKDD)*.

Flach, P. A., & Kull, M. (2015). Precision-recall-gain curves: PR analysis done right. *Advances in Neural Information Processing Systems (NIPS'15), 1*, 838–846.

Ghandeharioun, A., Eoff, B., Jou, B., & Picard, R. (2019). Characterizing sources of uncertainty to proxy calibration and disambiguate annotator and data bias. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, (pp. 4202–4206). IEEE.

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

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research, 13*(1), 723–773.

Guillory, D., Shankar, V., Ebrahimi, S., Darrell, T., & Schmidt, L. (2021). Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (pp. 1134–1144).

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *34th International Conference on Machine Learning*, (pp. 1321–1330), Sydney, Australia. PMLR.

Hagedorn, R., Doblas-Reyes, F. J., & Palmer, T. N. (2005). The rationale behind the success of multi-model ensembles in seasonal forecasting - I. Basic concept. *Tellus A: Dynamic Meteorology and Oceanography, 57*(3), 219–233.

Hallenbeck, C. (1920). Forecasting precipitation in percentages of probability. *Monthly Weather Review, 48*(11), 645–647.

Hinton, G., Vinyals, O., & Dean, J., et al. (2015). Distilling the knowledge in a neural network. *arXiv preprint*arXiv:1503.02531*, 2(7)*.

Hosmer, D., & Lemeshow, S. (2003). *Applied Logistic Regression*. New Jersey: Wiley.

Hüllermeier, E., Destercke, S., & Shaker, M. H. (2022). Quantification of credal uncertainty in machine learning: A critical analysis and empirical comparison. In J. Cussens, & K. Zhang (Eds.), *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, vol. 180 of *Proceedings of Machine Learning Research*, (pp. 548–557). PMLR.

Hüllermeier, E., & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning, 110*(3), 457–506.

Jiang, Y., Nagarajan, V., Baek, C., & Kolter, J. Z. (2021). Assessing generalization of sgd via disagreement. *arXiv preprint*arXiv:2106.13799*.*

Johansson, U., Löfström, T., Sundell, H., Linusson, H., Gidenstam, A., & Boström, H. (2018). Venn predictors for well-calibrated probability estimation trees. In A. Gammerman, V. Vovk, Z. Luo, E. Smirnov, & R. Peeters (Eds.), *Proceedings of the Seventh Workshop on Conformal and Probabilistic Prediction and Applications*, (pp. 3–14). PMLR.

Kängsepp, M., Valk, K., & Kull, M. (2022). On the usefulness of the fit-on-the-test view on evaluating calibration of classifiers. arXiv. Retrieved from arXiv:2203.08958, https://doi.org/10.48550/ARXIV.2203.08958

Koenker, R., & Hallock, K. F. (2001). Quantile regression. *Journal of Economic Perspectives, 15*(4), 143–156.

Kristiadi, A., Hein, M., & Hennig, P. (2020). Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International Conference on Machine Learning*, (pp. 5436–5446). PMLR.

Kuleshov, V., & Deshpande, S. (2022). Calibrated and sharp uncertainties in deep learning via density estimation. In *International Conference on Machine Learning*, (pp. 11683–11693). PMLR.

Kuleshov, V., Fenner, N., & Ermon, S. (2018). Accurate uncertainties for deep learning using calibrated regression. In *35th International Conference on Machine Learning, ICML 2018*, vol. 6, (pp. 4369–4377). International Machine Learning Society (IMLS).

Kuleshov, V., & Liang, P. S. (2015). Calibrated structured prediction. *Advances in Neural Information Processing Systems, 28*, 3474–3482.

Kull, M., & Flach, P. (2015). Novel decompositions of proper scoring rules for classification: Score adjustment as precursor to calibration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, (pp. 68–85). Springer.

Kull, M., Perello-Nieto, M., Kängsepp, M., Filho, T. S., Song, H., & Flach, P. (2019). Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with Dirichlet calibration. *Advances in Neural Information Processing Systems (NeurIPS'19), 32*, 12316–12326.

Kull, M., Silva Filho, T., & Flach, P. (2017a). Beta calibration: A well-founded and easily implemented improvement on logistic calibration for binary classifiers. In A. Singh, & J. Zhu (Eds.), In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, vol. 54 of *Proceedings of Machine Learning Research*, (pp. 623–631). Fort Lauderdale, FL, USA. PMLR, PMLR.

Kull, M., Silva Filho, T. M., & Flach, P. (2017). Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration. *Electronic Journal of Statistics, 11*(2), 5052–5080.

Kumar, A., Liang, P., & Ma, T. (2019). Verified Uncertainty Calibration. In: *Advances in Neural Information Processing Systems (NeurIPS'19)*.

Kumar, A., Sarawagi, S., &Jain, U. (2018). Trainable Calibration Measures For Neural Networks From Kernel Mean Embeddings. In Dy, J. and Krause, A., editors, *35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research*, vol. 80 of *Proceedings of Machine Learning Research*, (pp. 2805–2814). Stockholmsmässan, Stockholm Sweden. PMLR.

Lachiche, N., & Flach, P. A. (2003). Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, (pp. 416–423).

Leathart, T., Frank, E., Holmes, G., & Pfahringer, B. (2017). Probability Calibration Trees. In *Asian Conference on Machine Learning*, (pp. 145–160). PMLR.

Lichtenstein, S., Fischhoff, B., & Phillips, L. D. (1977). Calibration of probabilities: The state of the art. In *Decision Making and Change in Human Affairs*, (pp. 275–324). Springer.

Lichtenstein, S., Fischhoff, B., & Phillips, L. D. (1982). *Calibration of Probabilities: The state of the art to 1980* (pp. 306–334). Cambridge: Cambridge University Press.

Liu, J. Z., Paisley, J., Kioumourtzoglou, M.-A., & Coull, B. (2019). Accurate uncertainty estimation and decomposition in ensemble learning. *arXiv preprint*arXiv:1911.04061.

Liu, S., Long, M., Wang, J., & Jordan, M. I. (2018). Generalized zero-shot learning with deep calibration network. *Advances in Neural Information Processing Systems, 31*, 2005–2015.

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

Menon, A. K., & Williamson, R. C. (2018). A loss framework for calibrated anomaly detection. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, (pp. 1494–1504).

Milios, D., Michiardi, P., Rosasco, L., & Filippone, M. (2018). Dirichlet-based gaussian processes for large-scale calibrated classification. *Advances in Neural Information Processing Systems (NIPS'18), 31*, 6005–6015.

Miller, R. G. (1962). *Statistical Prediction by Discriminant Analysis* (pp. 1–54). Boston, MA: American Meteorological Society.

Müller, R., Kornblith, S., & Hinton, G. (2019). When does label smoothing help? *arXiv preprint*arXiv:1906.02629.

Murphy, A. H., & Winkler, R. L. (1977). Reliability of subjective probability forecasts of precipitation and temperature. *Journal of the Royal Statistical Society Series C (Applied Statistics), 26*(1), 41–47.

Naeini, M. P., & Cooper, G. F. (2016). Binary Classifier Calibration Using An Ensemble Of Near Isotonic Regression Models. In *IEEE 16th International Conference on Data Mining (ICDM)* (pp. 360–369). Institute of Electrical and Electronics Engineers (IEEE): IEEE.

Naeini, P., Cooper, G. F., & Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. In *29th AAAI Conference on Artificial Intelligence*, vol. 29.

Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. In *22nd International Conference on Machine Learning (ICML'05)*, (pp. 625–632), New York, New York, USA. ACM Press.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., & Snoek, J. (2019). Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems, 32*.

Patel, K., Beluch, W. H., Yang, B., Pfeiffer, M., & Zhang, D. (2021). Multi-class uncertainty calibration via mutual information maximization-based binning. In *International Conference on Learning Representations*.

Perello-Nieto, M., Song, H., Silva Filho, T., & Kängsepp, M. (2021). *Pycalib a library for classifier calibration*. Zenodo: Retrieved from. https://doi.org/10.5281/zenodo.5518877 (If you use this software, please cite it as below)

Platt, J. (2000). Probabilities for SV Machines. In A. J. Smola, P. Bartlett, B. Schölkopf, & D. Schuurmans (Eds.), *Advances in Large-Margin Classifiers* (pp. 61–74). Cambridge: MIT Press.

Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers* (pp. 61–74). Cambridge: MIT Press.

Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J., & Weinberger, K. Q. (2017). On fairness and calibration. *arXiv preprint*arXiv:1709.02012.

Roelofs, R., Cain, N., Shlens, J., & Mozer, M. C. (2021). Mitigating bias in calibration error estimation. *arXiv preprint*arXiv:2012.08668.

Romano, Y., Patterson, E., & Candes, E. (2019). Conformalized quantile regression. *Advances in Neural Information Processing Systems, 32*, 3543–3553.

Sahoo, R., Zhao, S., Chen, A., & Ermon, S. (2021). Reliable decisions with threshold calibration. *Threshold, 3*, 3–5.

Sanders, F. (1963). On subjective probability forecasting. *Journal of Applied Meteorology, 2*(2), 191–201.

Savage, L. J. (1971). Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association, 66*(336), 783–801.

Senge, R., Bösner, S., Dembczyński, K., Haasenritter, J., Hirsch, O., Donner-Banzhoff, N., & Hüllermeier, E. (2014). Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences, 255*, 16–29.

Shaker, M. H., & Hüllermeier, E. (2021). Ensemble-based uncertainty quantification: Bayesian versus credal inference. arXiv. Retrieved from arXiv:2107.10384, https://doi.org/10.48550/ARXIV.2107.10384

Song, H., Diethe, T., Kull, M., & Flach, P. (2019). Distribution calibration for regression. In Chaudhuri, K. and Salakhutdinov, R., editors, *36th International Conference on Machine Learning*, vol. 97, (pp. 5897–5906), Long Beach, California, USA. PMLR.

Thulasidasan, S., Chennupati, G., Bilmes, J., Bhattacharya, T., & Michalak, S. (2019). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *arXiv preprint*arXiv:1905.11001.

Tran, G.-L., Bonilla, E. V., Cunningham, J., Michiardi, P., & Filippone, M. (2019). Calibrating deep convolutional gaussian processes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, (pp. 1554–1563). PMLR.

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). PMLR.

Vovk, V., Gammerman, A., & Shafer, G. (2005). *Algorithmic learning in a random world*. Berlin: Springer Science & Business Media.

Vovk, V., & Petej, I. (2012). Venn-abers predictors. *arXiv preprint*arXiv:1211.0025.

Vovk, V., Shafer, G., & Nouretdinov, I. (2003). Self-calibrating probability forecasting. *Advances in Neural Information Processing Systems*, 16.

Wang, Y., Li, L., & Dang, C. (2019). Calibrating classification probabilities with shape-restricted polynomial regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 41*(8), 1813–1827.

Wen, Y., Tran, D., & Ba, J. (2020). Batchensemble: An alternative approach to efficient ensemble and lifelong learning. *arXiv preprint*arXiv:2002.06715.
Wenger, J., Kjellström, H., & Triebel, R. (2020). Non-parametric calibration for classification. In *International Conference on Artificial Intelligence and Statistics*, (pp. 178–190). PMLR.
Widmann, D., Lindsten, F., & Zachariah, D. (2019). Calibration tests in multi-class classification: A unifying framework. *Advances in Neural Information Processing Systems, 32*, 12257–12267.
Widmann, D., Lindsten, F., & Zachariah, D. (2022). Calibration tests beyond classification. arXiv:2210.13355, https://doi.org/10.48550/ARXIV.2210.13355
Winkler, R. L. (1969). Scoring rules and the evaluation of probability assessors. *Journal of the American Statistical Association, 64*(327), 1073–1078.
Zadrozny, B., & Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. *ICML, 1*, 609–616.
Zadrozny, B. & Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *8th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '02*, (pp. 694–699), New York, New York, USA. ACM Press.
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).

Authors and Affiliations

Telmo Silva Filho[1,2] · Hao Song[2] · Miquel Perello-Nieto[2] · Raul Santos-Rodriguez[2] · Meelis Kull[3] · Peter Flach[2]

✉ Miquel Perello-Nieto
miquel.perellonieto@bristol.ac.uk

Telmo Silva Filho
telmo.silvafilho@bristol.ac.uk

Hao Song
hao.song@bristol.ac.uk

Raul Santos-Rodriguez
enrsr@bristol.ac.uk

Meelis Kull
meelis.kull@ut.ee

Peter Flach
peter.flach@bristol.ac.uk

[1] Department of Statistics, Federal University of Paraíba, Cidade Universitária, João Pessoa, Paraíba 58.051-900, Brazil

[2] Intelligent Systems Laboratory, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol BS8 1UB, United Kingdom

[3] Department of Computer Science, University of Tartu, Ülikooli 18, Tartu 50090, Estonia