CORE

Metadata, citation and similar papers at core.ac.uk

Provided by Explore Bristol Research

OPEN ACCESS

University of BRISTOL

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. https://doi.org/10.1214/17-EJS1338SI

Peer reviewed version

License (if available):
CC BY

Link to published version (if available):
10.1214/17-EJS1338SI

Link to publication record in Explore Bristol Research
PDF-document

This is the final published version of the article (version of record). It first appeared online via Project Euclid at https://doi.org/10.1214/17-EJS1338SI . Please refer to any applicable terms of use of the publisher.

University of Bristol - Explore Bristol Research

General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available: http://www.bristol.ac.uk/pure/about/ebr-terms

Electronic Journal of Statistics
Vol. 11 (2017) 5052–5080
ISSN: 1935-7524
https://doi.org/10.1214/17-EJS1338SI

Beyond sigmoids: How to obtain well-calibrated probabilities from binary classifiers with beta calibration

Meelis Kull

Institute of Computer Science, University of Tartu, Estonia
e-mail: meelis.kull@ut.ee

Telmo M. Silva Filho

Center of Informatics, Federal University of Pernambuco, Brazil
e-mail: tmsf@cin.ufpe.br

and

Peter Flach

Department of Computer Science, University of Bristol, United Kingdom
e-mail: peter.flach@bristol.ac.uk

Abstract: For optimal decision making under variable class distributions and misclassification costs a classifier needs to produce well-calibrated estimates of the posterior probability. Isotonic calibration is a powerful non-parametric method that is however prone to overfitting on smaller datasets; hence a parametric method based on the logistic sigmoidal curve is commonly used. While logistic calibration is designed for normally distributed per-class scores, we demonstrate experimentally that many classifiers including Naive Bayes and Adaboost suffer from a particular distortion where these score distributions are heavily skewed. In such cases logistic calibration can easily yield probability estimates that are worse than the original scores. Moreover, the logistic curve family does not include the identity function, and hence logistic calibration can easily uncalibrate a perfectly calibrated classifier.

In this paper we solve all these problems with a richer class of parametric calibration maps based on the beta distribution. We derive the method from first principles and show that fitting it is as easy as fitting a logistic curve. Extensive experiments show that beta calibration is superior to logistic calibration for a wide range of classifiers: Naive Bayes, Adaboost, random forest, logistic regression, support vector machine and multi-layer perceptron. If the original classifier is already calibrated, then beta calibration learns a function close to the identity. On this we build a statistical test to recognise if the model deviates from being well-calibrated.

Keywords and phrases: Binary classification, classifier calibration, posterior probabilities, logistic function, sigmoid, beta distribution.

Received June 2017.

Contents

1 Introduction 5053
2 Logistic calibration 5056
2.1 Logistic family of calibration maps 5056
2.2 Fitting the logistic calibration maps 5058
3 Beta calibration 5059
3.1 Beta calibration from first principles 5059
3.2 Examples of beta calibration 5060
3.3 Fitting the parameters 5061
3.4 Further practical examples 5064
4 Experiments 5066
4.1 Experimental method 5067
4.2 Beta calibration versus isotonic and logistic calibration 5068
4.3 Parametric calibration methods 5069
4.4 The influence of dataset size on calibration performance 5070
5 Statistical beta calibration test 5072
5.1 Test statistic: dissimilarity from the identity map 5074
5.2 Approximating the null distributions with gamma distributions 5075
5.3 Algorithm and experiments 5077
6 Conclusions 5078
Acknowledgements 5079
References 5079

1. Introduction

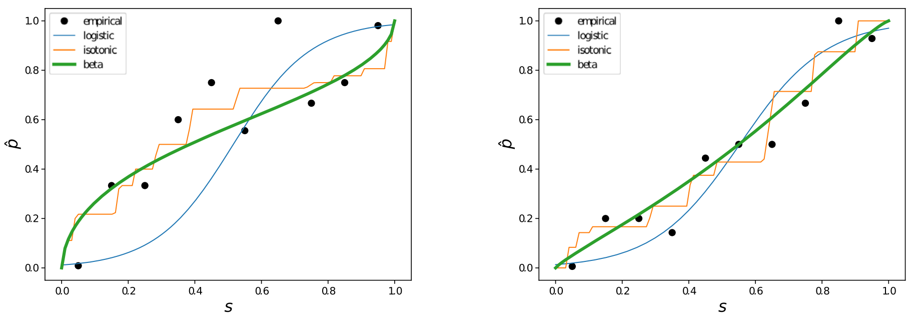
A predictive model can be said to be *well-calibrated* if its predictions match observed distributions in the data. In particular, a probabilistic classifier is well-calibrated if, among the instances receiving a predicted probability vector p , the class distribution is approximately distributed as p (Zadrozny and Elkan, 2002). Hence the classifier approximates, in some sense, the class posterior, although the approximation can be crude: for example, a constant classifier predicting the overall class distribution for every instance is perfectly calibrated in this sense. Calibration is closely related to optimal decision making and cost-sensitive classification, where we wish to determine the predicted class that minimises expected misclassification cost. The better our estimates of the class posterior are, the closer we get to the (irreducible) Bayes risk. The scores of a sufficiently calibrated classifier can be simply thresholded at a threshold directly derived from the misclassification cost (Zadrozny and Elkan, 2002). Thresholds can also be derived to optimally adapt to a change in class prior, or to a combination of both. In contrast, for a poorly calibrated classifier the optimal thresholds cannot be obtained without optimisation; for example, by means of ROC analysis (Provost and Fawcett, 2001).

Some learning algorithms are designed to yield well-calibrated probabilities. These include decision trees, whose leaf probabilities are optimal on the training set; as trees suffer from high variance, using Laplace smoothing and no pruning is recommended (Ferri, Flach and Hernández-Orallo, 2003). Logistic regression is another example of a learning algorithm that often produces well-calibrated probabilities; as we show in this paper, this only holds if the specific parametric assumptions made by logistic regression are met, which cannot be guaranteed in general. Many other learning algorithms do not take sufficient account of distributional factors (e.g., support vector machines) or make unrealistic assumptions (e.g., Naive Bayes) and need to be calibrated in post-processing. Well-established calibration methods in binary classification include logistic calibration, also known as 'Platt scaling' in reference to the author who introduced it for support vector machines (Platt, 2000) and isotonic calibration, also known as the ROC convex hull method and pool-adjacent-violators (Zadrozny and Elkan, 2002; Fawcett and Niculescu-Mizil, 2007).

Isotonic calibration is a non-parametric method to calibrate scoring classifiers in binary classification. It uses the convex hull of the model's ROC curve to discretise the scores into bins; the slope of each segment of the convex hull can be interpreted as an empirical likelihood ratio, from which a calibrated posterior probability for the corresponding bin can be derived. Since multiple scores can be mapped to the same bin, the resulting calibration map consists of a set of points that look like a staircase function, which can be interpolated if a continuous calibration map is needed (the Scikit-learn implementation we use in our experiments interpolates linearly). Isotonic calibration always provides monotonic calibration maps, meaning that it trusts the ranking produced by the original classifier and for any two instances assigns a higher (or same) calibrated probability of the positive class to the instance with higher original score. Figure 1 shows two examples with the isotonic calibration map in orange, where the x-axis represents the positive class probability as output by Adaboost and Naive Bayes on the landsat-satellite and vowel datasets and the y-axis shows the respective calibrated probability. Details about the experimental setup are provided in Section 4.

Logistic calibration is a parametric method with sigmoidal calibration maps which can be derived from the assumption that the scores within each class are normally distributed with the same variance. The logistic sigmoid is specified by two parameters: a location parameter m specifying the midpoint of the sigmoid at which the calibrated score is 0.5; and a shape parameter γ specifying the slope of the sigmoid at this midpoint. Figure 1 shows logistic calibration maps in blue. Being a parametric model, logistic calibration tends to require a smaller amount of labelled data to fit the sigmoid; however, it can produce bad results due to model mismatch. This mismatch can be substantial and can actually lead to 'calibrated' scores that are worse than the original. Figure 1 shows a case where the score distortions (black dots) are clearly not sigmoidal. Isotonic calibration (orange line) captures this but logistic calibration (blue line) results in a very poor fit.

The green line in Figure 1 shows that our proposed beta calibration method provides a similar fit to isotonic calibration on these datasets. Note that in the left figure the beta calibration map has an inverse-sigmoidal shape, which is outside of the logistic family of calibration maps. This is no coincidence:



(a) Adaboost – landsat-satellite (b) Naive Bayes – vowel

FIG 1. *Calibration maps obtained from logistic calibration, isotonic calibration and beta calibration as learned on the positive class probabilities output by (a) Adaboost on the landsat-satellite dataset; and (b) Naive Bayes on the vowel dataset. The x-axis represents the probabilities output by the classifier (Adaboost or Naive Bayes) and the y-axis shows the respective calibrated probability. The "empirical" dots show the actual proportion of positives in the test data, where instances have been split into 10 equal-width bins according to the probability output by the classifier. In both datasets logistic calibration makes matters worse compared to the original probabilities, on both log-loss and Brier score. Beta calibration and isotonic calibration are both better than original probabilities, with beta calibration being best in all measures except with log-loss in vowel.*

the inverse sigmoid is appropriate for classifiers that tend to produce extreme scores close to 0 or 1, such as Adaboost. The figure on the right is another case with a non-sigmoidal pattern in the uncalibrated scores. Here the Naive Bayes scores are quite well calibrated and beta calibration learns a map very close to the identity function. Logistic calibration produces a poor fit as the identity function is not a member of the logistic family.

The main contributions of this paper are (i) a demonstration that model mismatch is a real danger for logistic calibration for a range of widely used machine learning models and can make classifiers **less** calibrated; and (ii) the derivation of a new and richer parametric family which fixes this in a principled and flexible way. The outline of the paper is as follows. In Section 2 we discuss the logistic calibration method and its properties. Section 3 introduces our new beta calibration method. In Section 4 we report on a wide range of experiments showing that beta calibration is superior to logistic calibration and the preferred calibration method for smaller datasets. Section 5 proposes a statistical test that can reveal if a classifier is not calibrated by comparing the beta calibration map with the identity function. Section 6 concludes with a short discussion of the main results.

A previous, shorter version of this paper appeared as (Kull, Silva Filho and Flach, 2017). The most significant additions include additional experiments with logistic regression, random forest, multi-layer perceptron and support vector machine; an empirical investigation of the effect of dataset size on the performance of the various calibration methods; and the novel statistical test described in Section 5.

2. Logistic calibration

The aim of calibration in binary classification is to take an uncalibrated scoring classifier $s = f(x)$ and apply a calibration map μ on top of it to produce calibrated probabilities $\mu(f(x))$. Formally, a scoring classifier is perfectly calibrated on a dataset if for each of its output scores s the proportion of positives within instances with model output score s is equal to s (Cohen and Goldszmidt, 2004). Denoting the instances in the dataset by $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and their binary labels by $y_1, \ldots, y_n$, a model f is calibrated on this dataset if for each of its possible outputs $s_i = f(\mathbf{x}_i)$ the following holds:

$$s_i = \mathbb{E}[Y|f(X) = s_i]$$

where the random variables X, Y denote respectively the features and label of a uniformly randomly drawn instance from the dataset, and the labels $Y = 1$ and $Y = 0$ stand for a positive and negative, respectively. This expectation can be rewritten as follows ($I[\cdot]$ is the indicator function):

$$\mathbb{E}[Y|f(X) = s_i] = \frac{\sum_{j=1}^{n} y_j \cdot I[f(\mathbf{x}_j) = s_i]}{\sum_{j=1}^{n} I[f(\mathbf{x}_j) = s_i]}$$

For any fixed model and dataset there exists a uniquely determined calibration map $\mu(s_i) = \mathbb{E}[Y|f(X) = s_i]$ which produces perfectly calibrated probabilities on the given dataset. However, usually we do not want to learn perfect calibration maps on the training data, because these would overfit and would be far from being calibrated on the test data. For example, if the model f outputs a unique score for each training instance then the training-perfect calibration map would produce only the 0/1-probabilities $\mu(s_i) = y_i$, which would generalise poorly. An obvious solution would be to assume a parametric form of the calibration map, for which the logistic family is often used.

2.1. *Logistic family of calibration maps*

Logistic calibration was proposed by (Platt, 2000) to reduce overfitting by introducing a strong inductive bias which considers only calibration maps of the following form:

$$\mu_{\mathsf{logistic}}(s; \gamma, \delta) = \frac{1}{1 + 1/\exp(\gamma \cdot s + \delta)}$$

where γ, δ are real-valued parameters with $\gamma \geq 0$ to ensure that the calibration map is monotonically non-decreasing. Monotonicity is enforced assuming that higher model output scores suggest a higher probability to be positive. For easier interpretability we introduce the parameter $m = -\delta/\gamma$. This implies $\delta = -m\gamma$, yielding the following alternative parametrisation:

$$\mu_{\mathsf{logistic}}(s; \gamma, -m\gamma) = \frac{1}{1 + 1/\exp(\gamma \cdot (s - m))} \qquad (1)$$

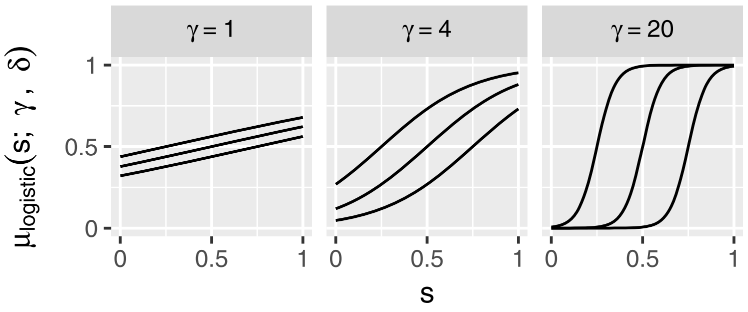


FIG 2. *Examples of logistic curves with parameters* $\gamma \in \{1, 4, 20\}$*,* $m \in \{0.25, 0.5, 0.75\}$ *and* $\delta = -m\gamma$*.*

The parameter m determines the value of s for which the calibrated score is $1/2$; the slope of the calibration map at $s = m$ is $\gamma/4$. Figure 2 shows a variety of shapes that the logistic calibration map can take.

We proceed to show that the parametric assumption made by logistic calibration is exactly the right one if the scores output by a classifier are normally distributed within each class around class means s^+ and s^- with the same variance σ^2. This gives class-specific probability density functions (PDFs)

$$
\begin{aligned}
p(s|+) &= C \exp[-(s - s^+)^2/(2\sigma^2)] \\
p(s|-) &= C \exp[-(s - s^-)^2/(2\sigma^2)]
\end{aligned}
$$

with $C = 1/\sqrt{2\pi}\sigma$, hence the likelihood ratio is

$$
\begin{aligned}
LR(s) =& \frac{p(s|+)}{p(s|-)} = \exp[(-(s - s^+)^2 + (s - s^-)^2)/(2\sigma^2)] \\
=& \exp[(2(s^+ - s^-)s - (s^{+2} - s^{-2}))/(2\sigma^2)] \\
=& \exp[(s^+ - s^-)/\sigma^2(s - (s^+ + s^-)/2)] \\
=& \exp[\gamma(s - m)]
\end{aligned}
$$

with $\gamma = (s^+ - s^-)/\sigma^2$ and $m = (s^+ + s^-)/2$. For $\gamma > 0$ this is a monotonically increasing function with $LR(m) = 1$.

We then derive a calibrated probability as follows:[1]

$$
\mu_{\mathsf{logistic}}(s; \gamma, -m\gamma) = \frac{1}{1 + LR(s)^{-1}} = \frac{1}{1 + \exp[-\gamma(s - m)]}
$$

giving the exact same form as in Eq.(1).

Conversely, it is easy to see that every function of this form corresponds to some pair of Gaussians with equal variance. Indeed, one can choose Gaussians

[1]Here we assume a uniform prior over the classes, hence the likelihood ratio equals the posterior odds. Adapting to a non-uniform prior can be done by moving the decision threshold on the calibrated probability away from 1/2.

with unit variance and with the means $s^+ = m + \gamma/2$ and $s^- = m - \gamma/2$ on positives and negatives, respectively. However, this is only one of the infinitely many possibilities. The same function corresponds to any pair of Gaussians with centres symmetrically around m if the distance between the centres is γ times the variance. This highlights that logistic regression does not model the score distributions on positives and negatives explicitly, but only cares about the distance between the centres of these distributions in units of variance.

Although logistic calibration uses exactly the right parametric form if the score distributions are Gaussian, there can be other distributions where it is exactly right as well. In other words, logistic calibration might also work well in non-Gaussian cases, as long as the ratio of the score distributions on positives and negatives behaves similarly to the ratio of Gaussians with equal variance.

2.2. Fitting the logistic calibration maps

In order to fit the parameters of logistic regression we need to decide how we measure the goodness of fit. That is, we need to measure how good the probability estimates $\hat{p}_i = \mu_{\mathsf{logistic}}(s_i; \gamma, \delta)$ are, given the actual labels y_i. A well-known method to evaluate estimates $\hat{p}_i$ is to use log-loss, which penalises predicting $\hat{p}_i$ for a positive instance with a loss of $-\ln \hat{p}_i$ and for a negative instance with a loss of $-\ln(1 - \hat{p}_i)$. To illustrate, the fully confident predictions $\hat{p}_i = 0$ and $\hat{p}_i = 1$ incur loss 0 if correct, and loss ∞ if wrong, whereas the least confident prediction $\hat{p}_i = 0.5$ has loss $\ln 2$ regardless of the correct label.

The overall log-loss can be expressed as follows:

$$LL(\hat{\mathbf{p}}, \mathbf{y}) = \sum_{i=1}^{n} y_i(-\ln \hat{p}_i) + (1 - y_i)(-\ln(1 - \hat{p}_i))$$

where $\hat{\mathbf{p}} = (\hat{p}_1, \dots, \hat{p}_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$. Log-loss can be rewritten as follows:

$$\begin{aligned} LL(\hat{\mathbf{p}}, \mathbf{y}) &= -\ln \prod_{i=1}^{n} \hat{p}_i^{y_i} (1 - \hat{p}_i)^{1 - y_i} \\ &= -\ln \left(\prod_{y_i = 1} \hat{p}_i \prod_{y_i = 0} (1 - \hat{p}_i) \right) \end{aligned}$$

which is the negative log-likelihood of the labels in the data. This implies that minimising log-loss is equivalent to maximising log-likelihood, which is a common fitting method known as maximum likelihood estimation (MLE).

In practice, the logistic calibration maps can be fitted by minimising log-loss using some gradient-based optimisation procedure, such as the "minimize" function provided by SciPy (Jones et al., 2001), which uses a quasi-Newton method to perform the optimisation. However, as the task is simply univariate logistic regression with feature $\mathbf{s}$ and label $\mathbf{y}$, it can be solved using the standard logistic regression functionality in any machine learning toolkit, including WEKA (Hall et al., 2009) and Scikit-learn (Pedregosa et al., 2011).

3. Beta calibration

We have shown that the parametric family in logistic calibration corresponds to cases with normal and homoscedastic distributions of scores within the classes. For probabilistic classifiers such as Naive Bayes such Gaussians are unreasonable due to their infinite support, while the classifier's scores are in the range $[0, 1]$. Hence it makes sense to derive an alternative parametric family of calibration functions using distributions with finite support, which is the central concept in this paper.

3.1. Beta calibration from first principles

A natural choice for score distributions in the range $[0, 1]$ is the beta distribution which has PDF

$$p(s; \alpha, \beta) = \frac{s^{\alpha-1}(1-s)^{\beta-1}}{B(\alpha, \beta)}$$

where $\alpha > 0$ and $\beta > 0$ are shape parameters and $B(\alpha, \beta)$ is the normalising beta function. So assume that the scores on both classes are beta-distributed with parameters α_0, β_0 and α_1, β_1, respectively. The likelihood ratio then becomes

$$\begin{aligned} LR(s; \alpha_0, \beta_0, \alpha_1, \beta_1) &= \frac{s^{\alpha_1-1}(1-s)^{\beta_1-1}}{B(\alpha_1, \beta_1)} \bigg/ \frac{s^{\alpha_0-1}(1-s)^{\beta_0-1}}{B(\alpha_0, \beta_0)} \\ &= \frac{s^{\alpha_1-1}(1-s)^{\beta_1-1}}{s^{\alpha_0-1}(1-s)^{\beta_0-1}} \bigg/ \frac{B(\alpha_1, \beta_1)}{B(\alpha_0, \beta_0)} \\ &= \frac{s^a}{(1-s)^b} \bigg/ K \end{aligned}$$

where $a = \alpha_1 - \alpha_0$, $b = \beta_0 - \beta_1$, and $K = B(\alpha_1, \beta_1)/B(\alpha_0, \beta_0)$. For later convenience we introduce an alternative parametrisation with $K = e^{-c}$:

$$LR(s; a, b, c) = \frac{s^a}{(1-s)^b} \bigg/ e^{-c}$$

As before, we can easily turn this likelihood ratio into a calibrated probability $\mu_{beta}(s; a, b, c) = 1/(1 + LR(s; a, b, c)^{-1})$, which finally gives the beta calibration map family:

$$\mu_{beta}(s; a, b, c) = \frac{1}{1 + 1 \Big/ \left(e^c \frac{s^a}{(1-s)^b}\right)}$$

We will require each calibration map to be monotonically non-decreasing, which implies $a, b \geq 0$.

Conversely, it is easy to see that every function of this form corresponds to some beta distributions for the scores on positives and negatives. To see this, consider any $a, b > 0$ and $c \in \mathbb{R}$ and fix $\alpha_0 = 1, \alpha_1 = 1 + a, \beta_0 = M + b, \beta_1 = M$

for some value $M \geq 0$. Then indeed $a = \alpha_1 - \alpha_0$ and $b = \beta_0 - \beta_1$, it remains to be shown that $B(\alpha_1, \beta_1)/B(\alpha_0, \beta_0) = e^{-c}$. For this note that the value of $B(x+a, y)/B(x, y+b)$ is 0 for $x = 1, y = 0$ but tends to ∞ when $x \to \infty$ while $y = 1$. Due to continuity there must exist values x, y for which this ratio equals e^{-c}.

Similarly to logistic calibration not strictly assuming Gaussians, beta calibration does not assume score distributions to be beta-distributed either. Beta calibration only assumes that the ratio of the score distributions on positives and negatives behaves similarly to the ratio of two beta distributions.

Although beta calibration requires classifier outputs to be between 0 and 1 it can still be applied to calibrate any scoring classifier. All that needs to be done is to first map the scores onto the range $[0, 1]$ using a fixed strictly monotonic transformation. In the experiments we will use the standard logistic function $\mu_{\mathsf{logistic}}(\cdot; 1, 0)$ for this purpose.

3.2. Examples of beta calibration

A simple but striking case where beta calibration gives exactly the right calibration map is the following. Suppose we have k features, each being a copy of the same feature. Suppose further that these features are perfectly calibrated in the sense that the model $f(\mathbf{x}) = f(x, \ldots, x) = x$ outputting the value of the first (or equivalently any other) feature is perfectly calibrated. Naive Bayes would consider these features independent and on an instance $\mathbf{x} = (x, \ldots, x)$ would output the score

$$s = \frac{x^k}{x^k + (1-x)^k}$$

This pushes the probability estimates towards the extremes and the perfect calibration map must bring these back to their original calibrated values x. Since

$$\frac{s}{1-s} = \left(\frac{x}{1-x}\right)^k$$

the perfect calibration map is

$$x = \frac{1}{1 + 1/\left(\frac{s^{1/k}}{(1-s)^{1/k}}\right)}$$

which belongs to the beta calibration family with the parameters $a = b = 1/k$ and $c = 0$. This means that beta calibration can correct for certain kinds of feature overweighting as might occur in Naive Bayes but also in boosting.

As another example, suppose that we do not know that the scores are already calibrated and we apply beta calibration. In such a case we would like the calibration procedure to learn the identity mapping. Since the identity function does not belong to the logistic family, logistic calibration would uncalibrate the scores. However, the beta calibration family does contain the identity function, parametrised by $a = b = 1$ and $c = 0$, and hence would keep the scores calibrated.

Similarly as for logistic calibration we can define a midpoint m such that $LR(m) = 1$, which gives $K = m^a/(1-m)^b$ and hence another alternative parametrisation

$$\begin{aligned} LR(s; a, b, c) &= LR(s; a, b, b\ln(1-m) - a\ln m) \\ &= \frac{s^a}{(1-s)^b} \Big/ \frac{m^a}{(1-m)^b} \end{aligned}$$

Figure 3 shows a variety of shapes that the beta calibration maps can take for different values of a (horizontally), b (vertically) and m (within each figure). The panels on the descending diagonal show cases where $a = b$; we refer to this as beta[a=b] calibration. On the bottom right we see the familiar sigmoidal shapes which are achieved with $a = b > 1$. The midpoint can be moved using m, but notice the curves are not translation-invariant as logistic sigmoids are, due to their finite support. On the top left we see pure inverse-sigmoidal curves $a = b < 1$ that are able to correct for extreme probabilities. The middle panel shows that the family includes the identity map on the diagonal, which can be pulled away in either direction by varying m, resulting in non-sigmoidal curves that would be appropriate if one class suffers from extreme scores while the other tends to be scored towards the middle. It can be shown that the beta[a=b] calibration family is closed under inversion: if $p = \mu_{beta}(s; a, a, c)$, then $s = \mu_{beta}(p; 1/a, 1/a, -c/a)$. This can be derived as follows:

$$\begin{aligned} \frac{p}{1-p} = \frac{\mu_{beta}(s; a, a, c)}{1 - \mu_{beta}(s; a, a, c)} &= e^c \frac{s^a}{(1-s)^a} \\ \left(e^{-c}\frac{p}{1-p}\right)^{\frac{1}{a}} &= \frac{s}{1-s} \\ \frac{\mu_{beta}(p; 1/a, 1/a, -c/a)}{1 - \mu_{beta}(p; 1/a, 1/a, -c/a)} &= \frac{s}{1-s}. \end{aligned}$$

The off-diagonal panels in Figure 3 show various asymmetries that can be introduced by allowing $a \neq b$. These asymmetries are strongest if one parameter is larger than 1 while the other is smaller than 1.

3.3. Fitting the parameters

One way of fitting beta calibration maps is to minimise log-loss with the same methods as in logistic regression. This can be performed using any optimisation tool, supplying it with the objective function and its gradient. However, in the following we derive results which reduce these tasks to fitting logistic regression in a different feature space, allowing to implement beta calibration by simply calling any logistic regression implementation, which is contained in all standard machine learning toolkits.

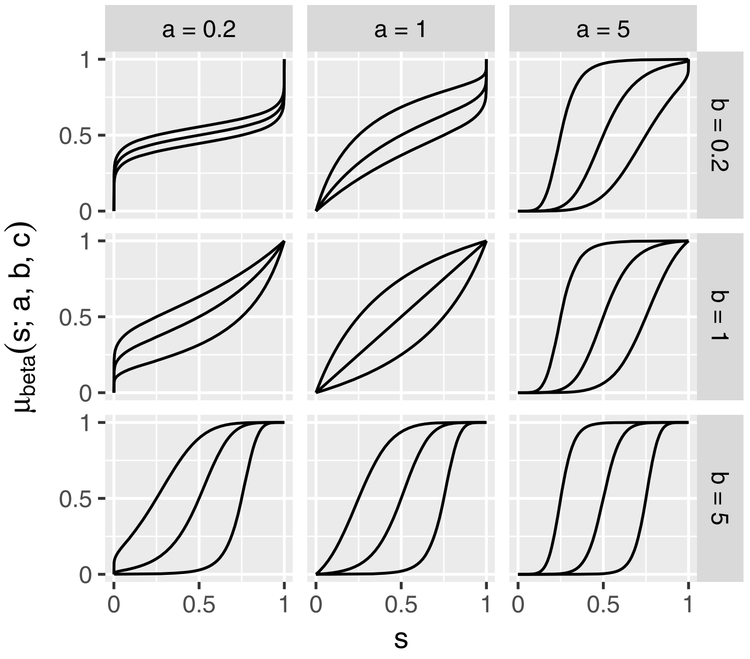


FIG 3. *Examples of beta curves with parameters* $a, b \in \{0.2, 1, 5\}$*,* $m \in \{0.25, 0.5, 0.75\}$ *and* $c = b\ln(1-m) - a\ln m$*.*

Proposition 1. *For any* $a \geq 0$ *and* $c, s \in \mathbb{R}$*:* $\mu_{\mathsf{beta}}(s; a, a, c) = \mu_{\mathsf{logistic}}(\ln \frac{s}{1-s}; a, c)$*.*

Proof. It is sufficient to prove that the corresponding likelihood ratios are equal.

$$\begin{aligned} LR_{\mathsf{logistic}}(\ln \frac{s}{1-s}; a, c) &= \exp[a \ln \frac{s}{1-s} + c] \\ &= \left(\frac{s}{1-s}\right)^a e^c = LR_{\mathsf{beta}}(s; a, a, c) \end{aligned} \qquad \square$$

The result in Proposition 1 shows that applying the beta calibration map with two parameters a and c where $b = a$ gives the same result as applying the logistic calibration map on the log-odds, i.e. on the log-ratios of scores and their complements, with $\gamma = a$ and $\delta = c$. Therefore, the optimal parameter values in minimising log-loss of beta-calibrated probabilities and in minimising log-loss of the logistic-calibrated log-odds-transformed scores coincide also. Hence, we can use logistic calibration (i.e. univariate logistic regression) to fit the beta[a=b] calibration maps, as shown in Algorithm 1. Note that the operators in the algorithm apply on vectors component-wise.

In a similar vein we can use bivariate logistic regression to fit the full 3-parameter beta calibration maps. This is due to our result stated in Proposition 2 showing that applying the beta calibration map with parameters a, b, c

Algorithm 1 Beta[a=b] calibration via logistic regression

```
Require: $\mathbf{y}_{\mathsf{train}}$ and $\mathbf{s}_{\mathsf{train}}$ are the label and model output score vectors on training instances,
    $\mathbf{s}_{\mathsf{test}}$ is the model output score vector on test instances
1: $\mathbf{s}' \leftarrow \ln \frac{\mathbf{s}_{\mathsf{train}}}{1-\mathbf{s}_{\mathsf{train}}}$
2: $(a, c) \leftarrow$ fit univariate logistic regression to predict $\mathbf{y}_{\mathsf{train}}$ from $\mathbf{s}'$
3: $\hat{\mathbf{p}}_{\mathsf{test}} \leftarrow 1/(1 + 1/(e^c \frac{\mathbf{s}_{\mathsf{test}}^a}{(1-\mathbf{s}_{\mathsf{test}})^a}))$
4: return $\hat{\mathbf{p}}_{\mathsf{test}}$
```

Algorithm 2 Beta calibration via logistic regression

```
Require: $\mathbf{y}_{\mathsf{train}}$ and $\mathbf{s}_{\mathsf{train}}$ are the label and model output score vectors on training instances,
    $\mathbf{s}_{\mathsf{test}}$ is the model output score vector on test instances
1: $\mathbf{s}' \leftarrow \ln \mathbf{s}_{\mathsf{train}}$
2: $\mathbf{s}'' \leftarrow -\ln(1 - \mathbf{s}_{\mathsf{train}})$
3: $(a, b, c) \leftarrow$ fit bivariate logistic regression to predict $\mathbf{y}_{\mathsf{train}}$ from $\mathbf{s}'$ and $\mathbf{s}''$
4: $\hat{\mathbf{p}}_{\mathsf{test}} \leftarrow 1/(1 + 1/(e^c \frac{\mathbf{s}_{\mathsf{test}}^a}{(1-\mathbf{s}_{\mathsf{test}})^b}))$
5: return $\hat{\mathbf{p}}_{\mathsf{test}}$
```

gives the same result as applying the bivariate logistic regression model on the log-scores and negative-log-complement-scores with the same parameters a, b, c. The resulting beta calibration algorithm is shown as Algorithm 2.

Proposition 2. *For any $a, b \geq 0$ and $c, s \in \mathbb{R}$:*

$$\mu_{\mathsf{beta}}(s; a, b, c) = \mu_{\mathsf{bilogistic}}(\ln s, -\ln(1-s); a, b, c),$$

where $\mu_{\mathsf{bilogistic}}(s', s''; a, b, c) = 1/(1+1/\exp[as'+bs''+c])$ is the bivariate logistic regression model family.

Proof. It is sufficient to prove that the corresponding likelihood ratios are equal.

$$\begin{aligned} & LR_{\mathsf{bilogistic}}(\ln s, -\ln(1-s); a, b, c) \\ & = \exp[a \ln s - b \ln(1-s) + c] \\ & = \frac{s^a}{(1-s)^b} e^c = LR_{\mathsf{beta}}(s; a, b, c) \end{aligned} \qquad \square$$

One subtle issue with Algorithms 1 and 2 is that fitting might result in either $a < 0$ or $b < 0$ or both, yielding calibration maps that are either monotonically decreasing or not monotonic at all. This did not ever happen in our experiments but if it is important to avoid this then logistic regression should be fitted with constraints $a, b \geq 0$. Alternatively, one could still use non-constrained logistic regression, but after this explicitly check if the constraints are satisfied. The variables with negative coefficients could then be eliminated (i.e., the respective coefficient fixed to be zero) and unconstrained logistic regression could be fitted again.

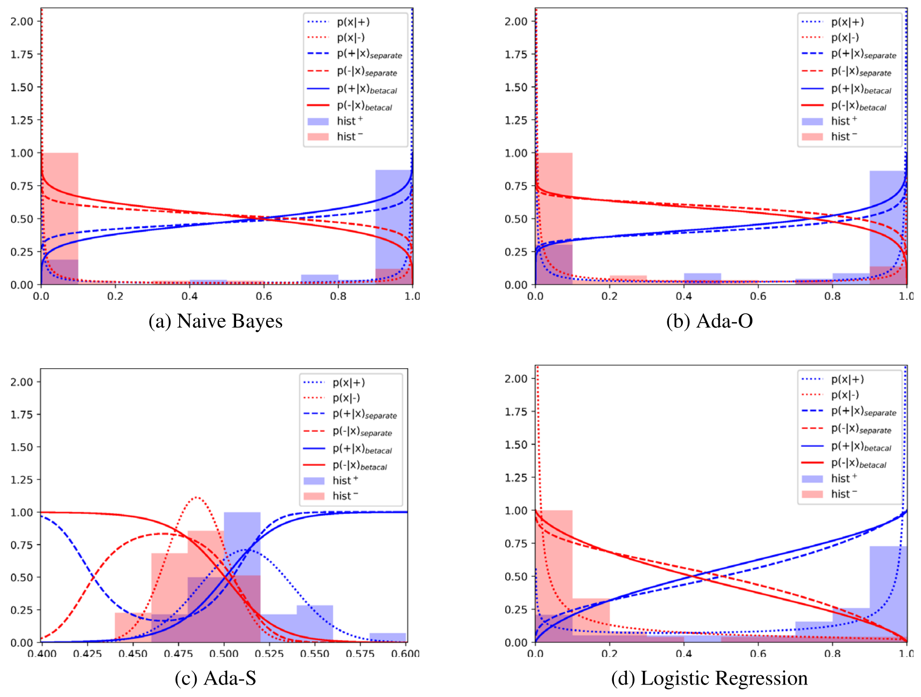
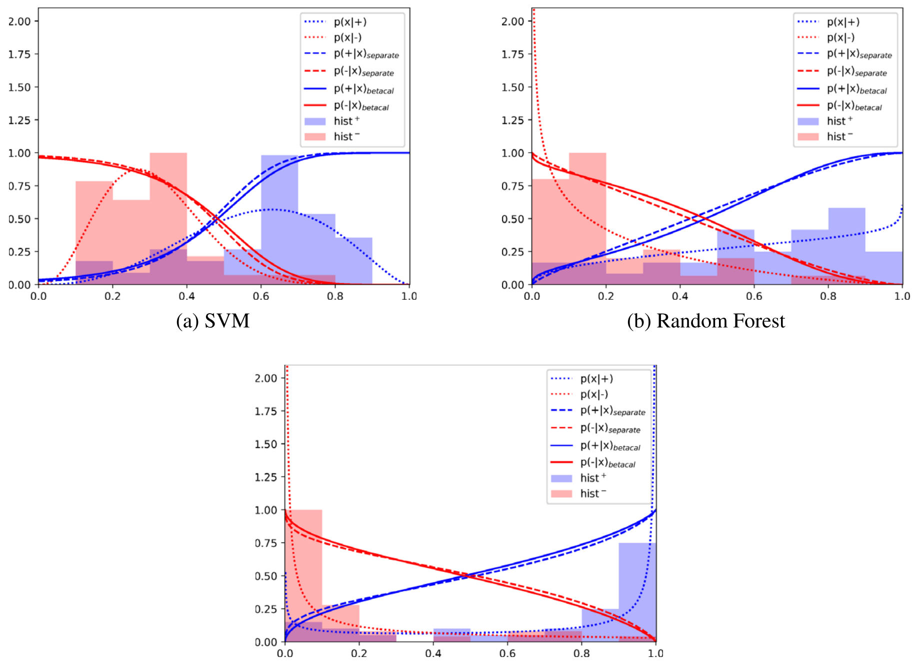


FIG 4. *Score distributions for Naive Bayes, Ada-O, Ada-S and Logistic Regression on the heart-statlog dataset. The blue and red histograms represent positive and negative scores, respectively. The dotted lines are beta distributions fitted using the method of moments. The dashed lines are class probabilities obtained from these distributions, and the solid lines show the calibration map learned by beta calibration.*

3.4. Further practical examples

We will now take a look at the distributions of the scores of seven base classifiers (Naive Bayes, Logistic Regression, SVM, Random Forest, MLP, and two versions of Adaboost) to get a visual idea of how well beta calibration fits them. Figures 4 and 5 present histograms of uncalibrated per-class scores (positives in blue and negatives in red) on the heart-statlog dataset. On these scores we fit per-class beta distributions using the method of moments, and show their densities with dotted lines using the same colours. For visualisation purposes, the histograms were normalized so that their maximum height is 1, and the densities were rescaled accordingly. We also plot two calibration maps, one obtained by beta calibration and shown with blue solid line, and the other by converting the two fitted beta distributions to posterior probabilities, shown with blue dashed line. For convenience we additionally plot the complements of these calibration maps in red, with the same line style.

This analysis looks at two versions of the Adaboost algorithm. The first is the original Adaboost with probabilities extracted in the standard way as in



(a) SVM

(b) Random Forest

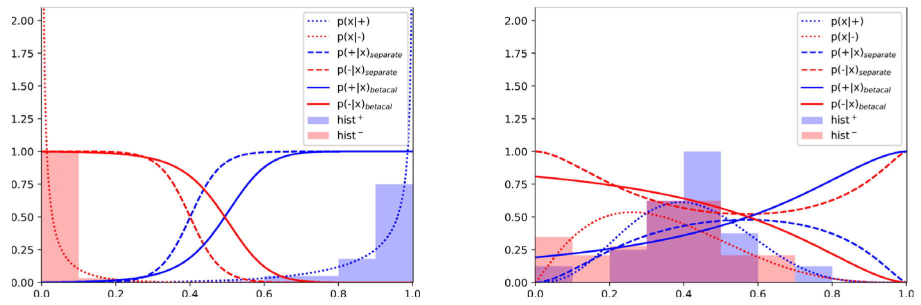
(c) MLP

Fig 5. *Score distributions for SVM, Random Forest and MLP on the heart-statlog dataset, continued from Figure 4.*

(Friedman, Hastie and Tibshirani, 2000), we refer to it as Ada-O. The second is the one implemented in Python's Scikit-learn, based on Adaboost method SAMME (Zhu et al., 2009), referred to as Ada-S. These versions turn out to be quite different. As Figures 4 and 5 show, Ada-O tends to push the probabilities to the extremes similarly to NB, LR and MLP, whereas Ada-S and SVM tend to pull them towards 0.5.

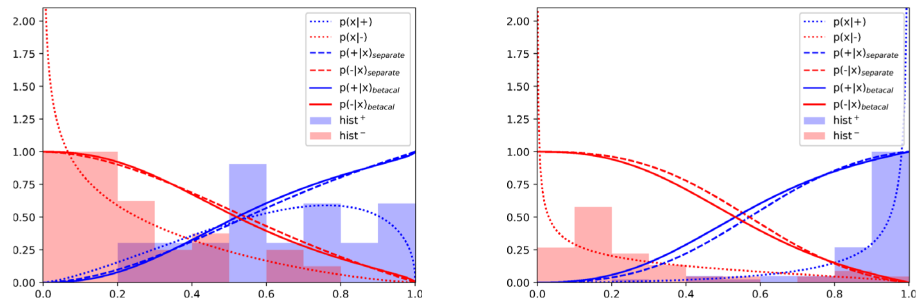
Some cases show bimodal beta distributions and we also see asymmetric distributions. In all cases, beta calibration and the two distributions found similar class probabilities, with some difference around the midpoint, which was expected, since the two distributions are fitted separately, while beta calibration tries to fit the class probabilities directly.

Figure 6 shows clear discrepancies between the class probabilities found by beta calibration and the two fitted beta distributions. In Figure 6a the difference lies mainly in the positions of the midpoints. In Figure 6b, according to the two fitted beta distributions, the positive class never shows higher class probabilities than the negative class. Therefore, the class probabilities are inverted on the right side of the figure. This would lead to a non-monotonically increasing calibration map. A similar situation can be seen on the left side of Figure 4c. In beta calibration we avoid this by setting any negative coefficient to zero and fitting the calibration model again.



(a) Logistic Regression - segment (b) Logistic Regression - glass

Fig 6. *Score distribution discrepancies for Logistic Regression. For the segment dataset, the posterior probabilities obtained from beta calibration and from the separate distributions have different midpoints. For the glass dataset, the right side of the figure shows a posterior probability inversion when obtained from separate distributions (dashed lines).*



(a) Random Forest - autos (b) Random Forest - ionosphere

Fig 7. *Score distributions of Random Forest for two datasets. The distributions fit by the method of moments for the negative scores do not match the respective histograms very well. Nevertheless, beta calibration and the separate distributions still obtained similar posterior probabilities.*

One important and useful side-effect of our parametrisation for beta calibration is that the scores do not need to be modelled by beta distributions, as long as the ratio of their distributions can be well modelled as the ratio of beta distributions. Figure 7 demonstrates this. In both cases, the beta distributions found by the method of moments for the negative scores do not match their respective histograms very well, which means that these distributions were not well fitted. Even so, beta calibration and the separate beta distributions find similar posterior probabilities, showing that for beta calibration it is enough to model the ratio of the distributions.

4. Experiments

We evaluated the effect of applying logistic, isotonic and beta calibration to the scores produced by Naive Bayes, Adaboost, Logistic Regression, Support Vector

Table 1
Description of the 41 classification datasets from UCI used for the experiments.

Name	Samples	Features	Classes
abalone	4177	8	3
autos	159	25	6
balance-scale	625	4	3
car	1728	6	4
cleveland	297	13	5
credit-approval	653	15	2
dermatology	358	34	6
diabetes	768	8	2
ecoli	336	7	8
flare	1389	10	6
german	1000	20	2
glass	214	9	6
heart-statlog	270	13	2
hepatitis	155	19	2
horse	300	27	2
ionosphere	351	34	2
iris	150	4	3
landsat-satellite	6435	36	6
letter	35000	16	26
libras-movement	360	90	15
lung-cancer	96	7129	2
mfeat-karhunen	2000	64	10
mfeat-morphological	2000	6	10
mfeat-zernike	2000	47	10
mushroom	8124	22	2
optdigits	5620	64	10
page-blocks	5473	10	5
pendigits	10992	16	10
scene-classification	2407	294	2
segment	2310	19	7
shuttle	101500	9	7
sonar	208	60	2
spambase	4601	57	2
tic-tac	958	9	2
vehicle	846	18	4
vowel	990	10	11
waveform-5000	5000	40	3
wdbc	569	30	2
wpbc	194	33	2
yeast	1484	8	10
zoo	101	16	7

Machine, Random Forest and Multi-Layer Perceptron on 41 datasets from UCI (Lichman, 2013), see Table 1 for details.

4.1. Experimental method

Multiclass datasets were transformed into two-class ones by taking the biggest class as positive and the remaining classes together as negative. We compared the performance of beta calibration, beta[a=b] calibration, beta[m=1/2] calibration, isotonic calibration, logistic calibration and uncalibrated probabilities, in terms of Brier score (BS) and log-loss (LL), which are both proper scoring rules (Kull and Flach, 2015) and hence well-founded measures for assessing the quality of predicted probabilities.

The results were obtained with 10 times 5-fold cross-validation, totalling 50 executions. Within each execution we used a 3-fold internal cross-validation with 2 folds for learning the model and 1 for fitting the calibration map. Thus, three calibrated classifiers were generated during each execution, the outputs of these three were averaged to provide predictions on the test fold. The same methodology was used in the paper proposing the logistic calibration method (Platt, 2000). All experiments were implemented in Python and the code is publicly available[2].

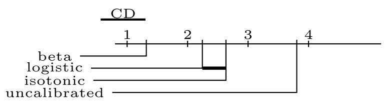
For Naive Bayes (NB), Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF) and Multi-Layer Perceptron (MLP), we used the implementations provided by Scikit-learn (Pedregosa et al., 2011) with default values for all parameters. Since SVM does not naturally output probabilities, we apply the standard sigmoid function $\mu_{\mathsf{logistic}}(\cdot; 1, 0)$ on its outputs to obtain values in the $[0, 1]$ range. For boosting we used 200 decision stumps as weak learners, which was the same number of trees used for RF.

For statistical comparisons between the calibration methods we followed (Demšar, 2006) and conducted the Friedman test based on the average ranks across the data sets to verify whether the differences between algorithms were statistically significant. In case of significance at 5% confidence level we proceeded to a post-hoc analysis based on Nemenyi statistics producing critical difference diagrams identifying pairwise significant differences.

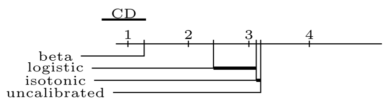
4.2. *Beta calibration versus isotonic and logistic calibration*

We first compared the full 3-parameter beta calibration with logistic and isotonic calibration methods, as well as with the uncalibrated probabilities, across all 7x2 settings (NB, Ada-S, Ada-O, LR, SVM, RF, MLP; LL, BS). The critical difference diagrams are shown in Figure 8 for LL and in Figure 9 for BS. It can be seen that beta calibration was significantly better than logistic calibration on NB, Ada-O, LR and MLP both for LL and BS. Furthermore, on LL beta calibration was even significantly better than isotonic calibration, while performing comparably on BS. With Ada-S, SVM the probabilities tend to be pulled towards 0.5 and the logistic fits reasonably well, here beta calibration performed comparably to logistic calibration (beta calibration was non-significantly better). To summarise, no other method was ever significantly better than beta calibration, and beta calibration was significantly better than all other methods for NB, Ada-O, LR and MLP according to LL. Full results on Ada-O for LL are shown in Table 2.

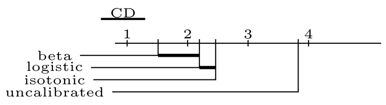
[2]https://betacal.github.io



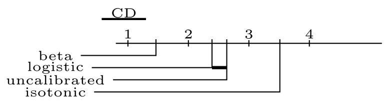
(a) Naive Bayes



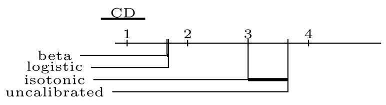
(b) Ada-O



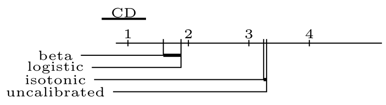
(c) Ada-S



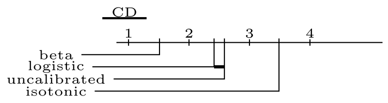
(d) Logistic Regression



(e) SVM



(f) Random Forest



(g) MLP

Fig 8. *Critical difference diagrams for log-loss with Naive Bayes, Ada-O, Ada-S, Logistic Regression, SVM, Random Forest and MLP as base classifiers, Friedman test p-values are* $6.9e{-}17$*,* $1.0e{-}12$*,* $4.7e{-}15$*,* $2.5e{-}11$*,* $5.7{-}17$*,* $3.2e{-}14$ *and* $1.3e{-}09$ *respectively.*

4.3. Parametric calibration methods

We continued to compare the variants of beta calibration against logistic calibration. The critical difference diagrams for this analysis are shown in Figure 10 for LL and in Figure 11 for BS. Among the three variants the 3-parameter version was either the best or tied with the best (i.e., not significantly worse than the best) for all settings (NB, Ada-O, Ada-S, LR, SVM, RF, MLP and LL, BS). Hence the experiments show that the full 3-parameter version of beta calibration is a versatile parametric calibration method which is preferable to logistic calibration, especially if the model has pushed the scores towards the extremes.

Beta[a=b] calibration has comparable running time to logistic calibration while the 3-parameter version can be slightly slower. However, both calibration methods are usually orders of magnitude faster than learning the classifier itself. Therefore, the overall time for learning a classifier and calibrating it is not increased considerably when moving from logistic calibration to beta calibration.

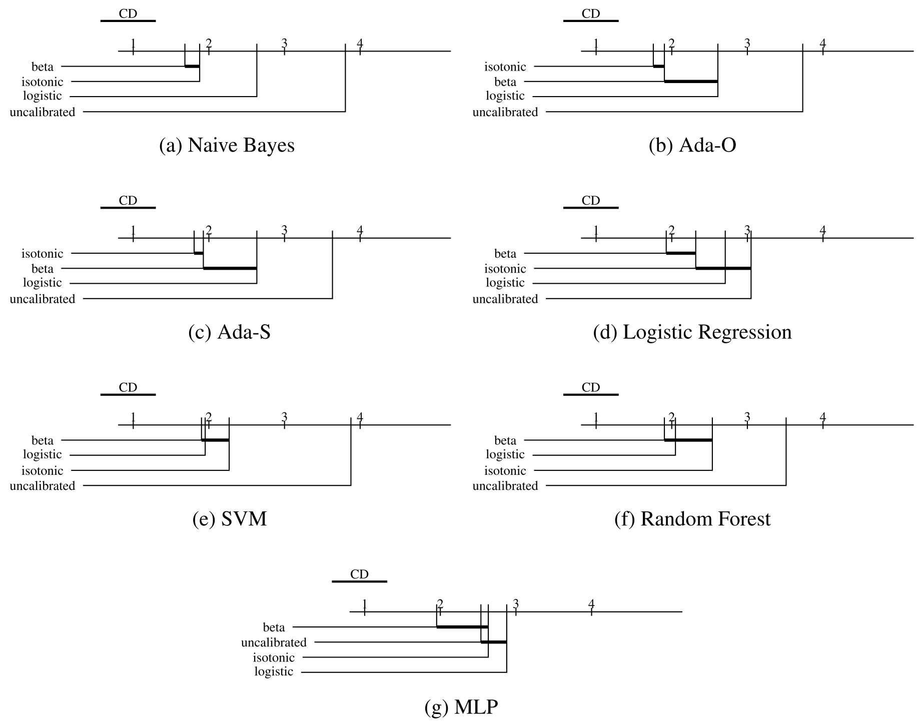


FIG 9. *Critical difference diagrams for Brier score with Naive Bayes, Ada-O, Ada-S, Logistic Regression, SVM, Random Forest and MLP as base classifiers, Friedman test p-values are* $1.0e{-14}$*,* $3.7e{-06}$*,* $5.8e{-13}$*,* 0.0006*,* $1.4e{-16}$*,* $1.6e{-09}$ *and* 0.0167 *respectively.*

4.4. *The influence of dataset size on calibration performance*

We evaluated the impact of dataset size on the performance of calibration methods by calculating the correlation between dataset size and each method's rank according to the loss measure (LL or BS) for each base classifier. Here, a negative correlation means that the bigger the dataset, the lower the method's rank, i.e., the method performed better relative to other methods.

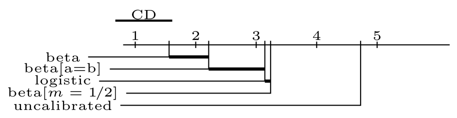
Tables 3 and 4 show the correlations for log-loss and Brier score, respectively. Isotonic calibration is negatively correlated in every base classifier for log-loss and in all but one classifier (Ada-S) for Brier score, showing that it usually benefits more from larger datasets than other methods. This was expected, since isotonic calibration is the only non-parametric method in this study and literature has shown that it needs more data than logistic calibration to fit a good calibration map (Niculescu-Mizil and Caruana, 2005).

To summarise our experimental analysis, we have seen that beta calibration works well on the scores of a variety of classifiers with different characteristics. In terms of log-loss and Brier score, beta calibration was never the worst calibra-

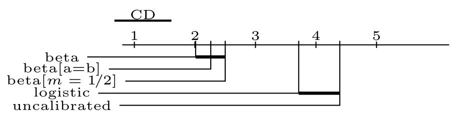
Table 2
*Log-loss results with Ada-O classifier. Best results are marked in **bold** and subscripts indicate the ranks (before rounding to 3 decimal digits).*

dataset	uncalibrated	beta	isotonic	logistic
abalone	0.612_2	$\mathbf{0.612}_1$	0.626_4	0.614_3
autos	0.427_4	$\mathbf{0.283}_1$	0.416_3	0.287_2
balance	0.049_4	$\mathbf{0.037}_1$	0.041_3	0.038_2
car	0.114_2	$\mathbf{0.109}_1$	0.122_4	0.119_3
clevela	0.496_3	$\mathbf{0.417}_1$	0.523_4	0.429_2
credit-	0.360_3	0.341_2	0.444_4	$\mathbf{0.338}_1$
dermato	0.036_4	$\mathbf{0.019}_1$	0.035_3	0.021_2
diabete	0.506_3	$\mathbf{0.484}_1$	0.525_4	0.495_2
ecoli	0.326_4	$\mathbf{0.145}_1$	0.224_3	0.159_2
flare	$\mathbf{0.404}_1$	0.404_2	0.421_4	0.408_3
german	0.511_3	0.506_2	0.538_4	$\mathbf{0.506}_1$
glass	0.696_4	0.489_2	0.560_3	$\mathbf{0.485}_1$
heart-s	0.570_4	$\mathbf{0.427}_1$	0.557_3	0.443_2
hepatit	0.805_4	$\mathbf{0.392}_1$	0.431_3	0.411_2
horse	0.642_4	$\mathbf{0.413}_1$	0.524_3	0.418_2
ionosph	0.381_4	$\mathbf{0.203}_1$	0.296_3	0.227_2
iris	0.127_4	0.000_2	$\mathbf{0.000}_1$	0.000_3
landsat	0.041_2	$\mathbf{0.040}_1$	0.043_3	0.053_4
letter	0.037_3	0.035_2	$\mathbf{0.035}_1$	0.044_4
libras-	0.589_4	$\mathbf{0.100}_1$	0.184_3	0.112_2
lung-ca	0.193_4	$\mathbf{0.139}_1$	0.189_3	0.139_2
mfeat-k	0.062_4	$\mathbf{0.027}_1$	0.048_3	0.032_2
mfeat-m	0.040_4	$\mathbf{0.014}_1$	0.026_3	0.014_2
mfeat-z	0.095_4	$\mathbf{0.032}_1$	0.063_3	0.039_2
mushroo	0.000_4	0.000_3	0.000_2	$\mathbf{0.000}_1$
optdigi	0.035_2	$\mathbf{0.033}_1$	0.044_4	0.040_3
page-bl	0.093_2	$\mathbf{0.089}_1$	0.098_3	0.109_4
pendigi	0.017_2	$\mathbf{0.017}_1$	0.023_4	0.021_3
scene-c	0.384_4	$\mathbf{0.364}_1$	0.376_2	0.378_3
segment	0.020_3	$\mathbf{0.010}_1$	0.023_4	0.013_2
shuttle	0.000_2	$\mathbf{0.000}_1$	0.001_4	0.000_3
sonar	0.941_4	$\mathbf{0.404}_1$	0.486_3	0.440_2
spambas	0.162_2	$\mathbf{0.162}_1$	0.173_4	0.167_3
tic-tac	0.379_4	$\mathbf{0.333}_1$	0.340_3	0.339_2
vehicle	0.077_2	$\mathbf{0.068}_1$	0.131_4	0.077_3
vowel	0.076_2	$\mathbf{0.071}_1$	0.094_4	0.085_3
wavefor	0.253_2	$\mathbf{0.252}_1$	0.264_3	0.271_4
wdbc	0.256_4	$\mathbf{0.089}_1$	0.141_3	0.107_2
wpbc	1.016_4	0.500_2	$\mathbf{0.496}_1$	0.503_3
yeast	0.510_2	$\mathbf{0.509}_1$	0.543_4	0.514_3
zoo	0.132_4	0.013_3	$\mathbf{0.013}_1$	0.013_2
rank	3.20	1.27	3.12	2.41

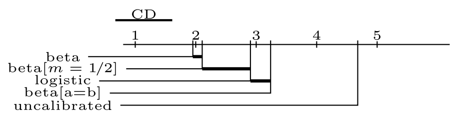
tion method, while frequently being the best one. Additionally, beta calibration seems able to fit good calibration maps regardless of dataset size, therefore it is a good alternative to isotonic calibration on smaller datasets, while generally outperforming logistic calibration.



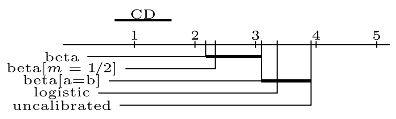
(a) Naive Bayes



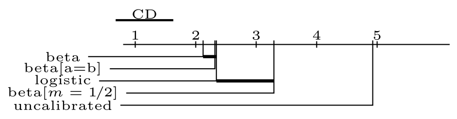
(b) Ada-O



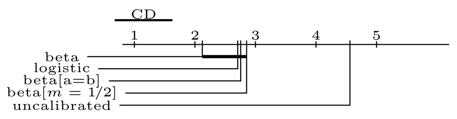
(c) Ada-S



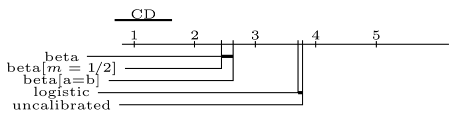
(d) Logistic Regression



(e) SVM



(f) Random Forest



(g) MLP

FIG 10. *Critical difference diagrams for log-loss on parametric methods with Naive Bayes, Ada-O, Ada-S, Logistic Regression, SVM, Random Forest and MLP classifiers, Friedman test p-values are* $1.2e{-}20$, $1.6e{-}15$, $2.8e{-}17$, $1.2{-}07$, $1.7e{-}20$, $2.2e{-}12$ *and* $1.9e{-}13$.

5. Statistical beta calibration test

It has already been mentioned that the parametric family of beta calibration maps includes the identity function. The benefit of this becomes apparent when beta calibration is applied to a classifier that is already calibrated. If choosing the identity mapping, beta calibration can leave the class probabilities unchanged. Conversely, if the learned calibration map is significantly different from the identity, the original classifier must have been poorly calibrated. This property inspired us to develop a calibration test based on beta calibration. Our test takes as input a particular labelled dataset with the class probabilities as predicted by a learned classifier, and outputs a p-value indicating how unlikely such situation is to occur if the classifier were calibrated.

The idea of the proposed test is to learn a beta calibration map and then measure how close the learned calibration map is to the identity. If it is sufficiently different, then it is highly unlikely that the original classifier is well-calibrated. However, if the learned map is very similar to the identity, then we have no information to reject the null hypothesis of the classifier being calibrated.[3]

[3]It is possible in this case that the true calibration map is far from the identity, while in

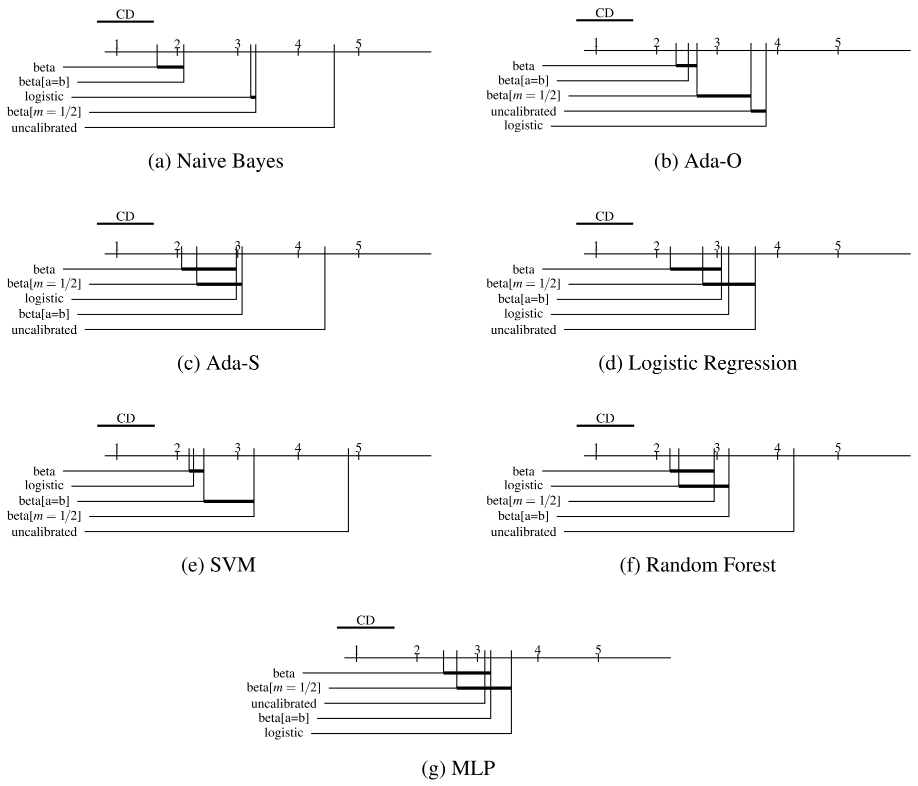


FIG 11. *Critical difference diagrams for Brier score on parametric methods with Naive Bayes, Ada-O, Ada-S, Logistic Regression, SVM, Random Forest and MLP classifiers, Friedman test p-values are* $2.4e{-}18$, $1.4e{-}06$, $1.6e{-}13$, 0.0006, $1.1e{-}20$, $3.6e{-}10$ *and* 0.0029.

TABLE 3
Correlation between method rank and dataset size for log-loss.

classifier	beta	beta[$m = 1/2$]	beta[a=b]	isotonic	logistic
NB	0.058	−0.035	0.252	−0.459	0.289
LR	−0.109	0.097	0.305	−0.510	0.140
Ada-O	−0.261	0.063	0.065	−0.067	0.167
Ada-S	0.033	−0.195	0.000	−0.012	0.246
RF	−0.271	−0.062	0.022	−0.058	0.277
MLP	0.021	0.205	0.315	−0.557	−0.099
SVM	0.191	0.205	0.245	−0.527	0.033

Beta calibration test can become useful in various scenarios:

- for any particular existing classifier it could suggest whether there is value in calibrating it with beta calibration;
- more generally, it could suggest whether beta calibration should be added to a particular machine learning pipeline;

the quite restrictive beta family the optimal calibration map is near the identity.

TABLE 4
Correlation between method rank and dataset size for Brier score.

classifier	beta	beta[$m = 1/2$]	beta[a=b]	isotonic	logistic
NB	0.308	−0.099	−0.099	−0.168	0.199
LR	−0.101	0.110	0.196	−0.248	0.062
Ada-O	0.053	0.107	0.363	−0.131	−0.305
Ada-S	−0.218	0.053	−0.180	0.165	0.051
RF	0.116	0.160	0.173	−0.262	−0.180
MLP	0.079	0.179	0.282	−0.356	−0.141
SVM	0.290	0.147	−0.003	−0.293	−0.072

- conversely, it could suggest whether calibration could be omitted from a pipeline;
- if the context has changed or drifted since the model or its calibration map was learned, then it could suggest whether a new calibration map should be learned for the new context.

In the following we will first define our test statistic, and next provide an algorithm to sample from the null distribution, enabling estimation of p-values and significance empirically. We will then demonstrate that the null distribution can be well approximated with a Gamma distribution, where the parameters can be easily estimated from the number of instances in the calibration fold. Finally, we will present the beta calibration test algorithm and discuss some experimental results on statistical calibration testing.

5.1. Test statistic: dissimilarity from the identity map

First, we define a test statistic that quantifies how different the learned calibration map is from the identity. One option would be to quantify the difference between the predicted probabilities before and after calibration on the instances from the given dataset. This would put a higher weight on the regions of the calibration map which are covered by more instances. Instead, we propose the test statistic that measures the area between the learned calibration map and the identity map. Formally, the test statistic A is defined as

$$A = \int_0^1 |\mu_{beta}(s; a, b, c) - s| \, ds$$

The advantage of not using actual instances in the formula is that it becomes easier to approximate the null distribution, as seen later.

The next step is to define the null distribution, that is the distribution of the area A assuming that the classifier is already perfectly calibrated. This would mean that every positive class probability $\hat{p}$ output by this classifier correctly reflects the proportion of positives among all instances with this prediction $\hat{p}$. Our approach to modelling the null distribution is to fix a given or imaginary dataset with predicted probabilities $\hat{p}_1, \ldots, \hat{p}_n$ and define the null distribution as areas generated in the following manner:

1. generate labels for each instance independently, where $y_i = 1$ with probability $\hat{p}_i$ and otherwise $y_i = 0$;
2. learn a calibration map to calibrate predictions $\hat{p}_1, \dots, \hat{p}_n$ as if $y_1, \dots, y_n$ were the corresponding true labels;
3. calculate the area between the obtained calibration map and the identity map.

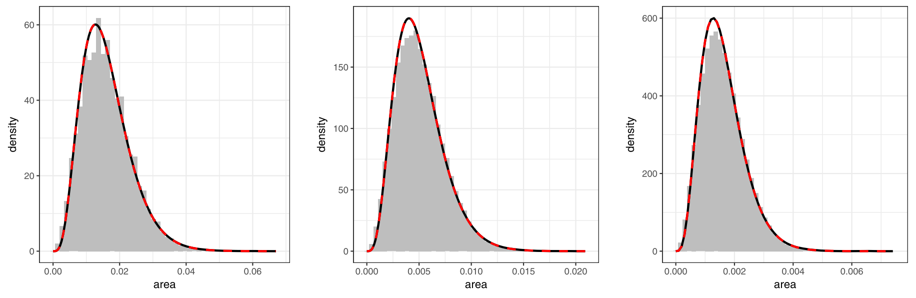
Again, this could be done on the given dataset, but we propose to do it on the imaginary dataset with equidistant uniform probabilities $\hat{p}_i = i/(n+1)$, for $i = 1, \dots, n$. The main advantage of this choice is to make the null distribution dependent only on the size n of the dataset, but not on the dataset itself. This allows to precompute the null distribution empirically for a range of different dataset sizes and use the results as lookup tables for p-values. Suppose we generated a big number of areas corresponding to size n in the precomputation phase. Then the estimated p-value for an area A between the identity and the calibration map from an actual dataset is equal to q if proportion q of the precomputed areas are larger than A and proportion $1 - q$ are smaller.

5.2. *Approximating the null distributions with gamma distributions*

The above approach requires storing all the precomputed areas for a range of different dataset sizes in order to later calculate the p-value for any given case. This can be a significant burden and therefore we investigated possibilities to approximate the distribution of areas with some known parametric family of distributions. It turned out that a gamma distribution provides a very good fit across a wide range of dataset sizes n. Figure 12 presents the histograms of the sampled null distributions for $n = 10^3, 10^4, 10^5$ and the corresponding gamma distributions fitted with the method of moment matching shown as the black solid line behind the red dashed line. In order to evaluate the goodness of fit we have plotted the corresponding quantile-quantile plots in Figure 13, see the black points behind the red points. The plots show that the gamma distributions fit very well but have slightly heavier tails than the empirical distribution. As a result, in the low p-value range the p-values from the fitted gamma distribution over-estimate the actual p-values. Due to this the gamma-approximated test gives significance less frequently. However, near the usual p-value significance thresholds like $p = 0.050$, $p = 0.010$ and $p = 0.001$ the difference is very small, e.g. for size $n = 10^5$ the corresponding p-value estimates from the gamma distribution are $p = 0.052$, $p = 0.013$ and $p = 0.002$.

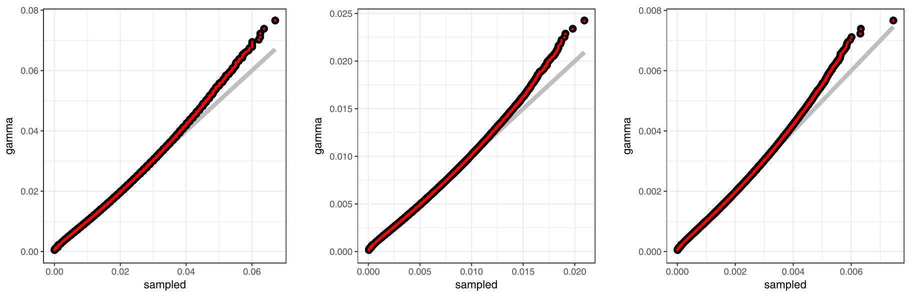
With the above approximation one would need to store only the parameters of the fitted gamma distributions for a wide range of dataset sizes. However, this is not ideal either, as it still requires a big precomputed table. We therefore derived simple formulas to approximate the parameters for the moment matching gamma distribution given the dataset size. In particular, the shape parameter k can be taken to be the following constant:

$$k = 4.84 \tag{2}$$



(a) size=1000 (b) size=10000 (c) size=100000

Fig 12. *The empirical and approximated null distributions for beta calibration test with three sizes of datasets. The empirical distribution of areas between the beta calibration map and the identity function has been obtained from 1 million imaginary datasets and is shown as the grey histogram. The black line represents the Gamma distribution fitted with moment matching and the red dashed line the Gamma distribution with parameters calculated with Eqs.(2) and (3).*



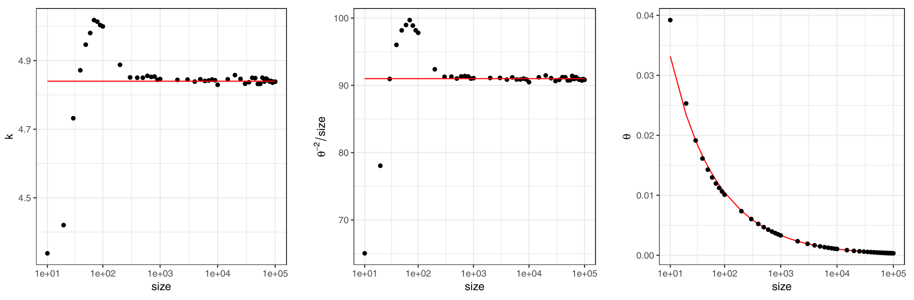
(a) size=1000 (b) size=10000 (c) size=100000

Fig 13. *Quantile-quantile plots comparing the empirical (x-axis) and fitted distributions (y-axis) from Figure 12. The black large dots correspond to the Gamma distribution obtained from mean matching and the red small dots to the Gamma distribution obtained with Eqs.(2) and (3).*

for all dataset sizes except very small ones, see Figure 14a. For the scale parameter θ we noticed that θ^{-2} is proportional to the dataset size (see Figure 14b), and hence θ can be approximated well as follows:

$$\theta = 1/\sqrt{91n} \tag{3}$$

(see Figure 14c). Figure 12 shows very good fit between the gamma distributions from moment matching with black line and from the approximation formulas with the red dashed line. The quantile-quantile plots in Figure 13 show that the p-values from the approximation formulas are very close to the p-values estimated using mean matching, also shown with red and black, respectively.



(a) Parameter k (b) Ratio of θ^{-2}/n (c) Parameter θ

FIG 14. *The values of parameters k and θ of the Gamma distributions fitted with mean matching for 46 different dataset sizes n ranging from 10 up to 100000: (a) fitted value of k and the constant regression line at 4.84; (b) the ratio θ^{-2}/n with fitted value of θ and the constant regression line at 91; (c) the fitted value of θ and the regression line $\theta = 1/\sqrt{91n}$.*

Algorithm 3 Beta calibration test p-values

```
Require: dataset with predicted probabilities $\hat{p}_1, \ldots, \hat{p}_n$ and actual labels $y_1, \ldots, y_n$
1: run beta calibration to obtain the parameters $a, b, c$ for the calibration map $\mu_{beta}(s; a, b, c)$
2: calculate the area $A$ between the calibration map and the identity map
3: calculate the p-value as the quantile of $A$ within the distribution $Gamma(4.84, 1/\sqrt{91n})$
4: return p-value
```

5.3. Algorithm and experiments

To summarise, the algorithm to obtain p-values from the beta calibration test is presented in Algorithm 3. Here the area A can be calculated by the standard trapezoid approximation, evaluating the calibration map at many equidistant uniform values between 0 and 1. The quantiles of the gamma distribution can be calculated by many statistical software toolkits, for example, with the command 'pgamma' in R.

We applied the developed statistical test on all SVM, LR, NB and MLP classifiers obtained in our experimental setting as described in Section 4. We split the classifiers into 6 bins according to the negative logarithm of the p-value that they received from our beta calibration test. The bin boundaries were $-1, \ldots, -5$, i.e. the respective p-value boundaries were $e^{-1} \approx 0.368, e^{-2} \approx 0.135, e^{-3} \approx 0.050, e^{-4} \approx 0.018, e^{-5} \approx 0.007$. Figure 15 shows in each bin the proportion of classifiers which improved after beta calibration. For p-values above 0.135 (neg-log-p-value below 2) the proportion of improvement is only slightly higher than 50%, meaning that whether beta calibration improves or not is quite random. For lower p-values (neg-log-p-value above 2) beta calibration improves (reduces log-loss) in increasingly higher proportion of cases.

Figure 16 presents the p-values per classifier and per dataset, with the grey-scale colours representing the same p-value bins as in Figure 15. The results show that MLP and LR tend to be more calibrated than SVM and NB, while there are several datasets where all classifiers are significantly non-calibrated.

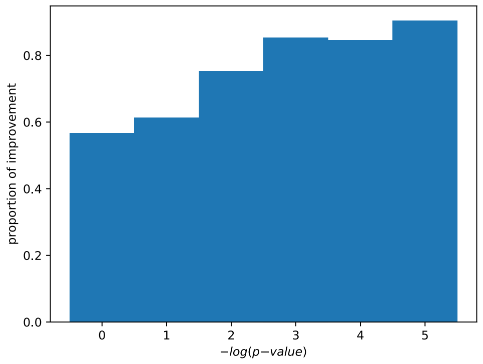


FIG 15. *The proportion of classifiers for which log-loss decreased after beta calibration, across all SVM, LR, NB and MLP classifiers from Section 4, grouped by negative logarithm of the p-value that they received from the beta calibration test.*

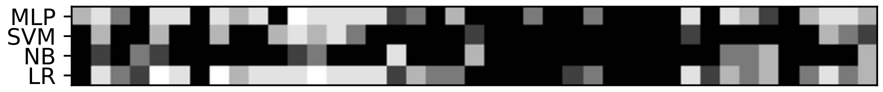


FIG 16. *The beta calibration test p-values of MLP, SVM, NB and LR (4 rows) per dataset (41 columns). Each p-value has been averaged over all cross-validation folds. The greyscale colours represent the same bins as in Figure 15, with white standing for negative logarithm p-value below 1 and black for above 5 (very significantly not calibrated). The order of datasets in the columns is the same as in Table 1.*

6. Conclusions

We introduced a rich and flexible family of calibration maps that includes both sigmoids and inverse sigmoids, as well as a host of other maps including the identity map. We derived the method from first principles making one simple parametric assumption: that the per-class scores of the classifier each follow a beta distribution. This is particularly suitable for classifiers that score on a bounded scale, for which the Gaussian assumption underlying logistic calibration is incoherent. The two beta distributions can be quite different in shape, giving the family much more flexibility than the logistic family, which has to assume homoscedasticity to keep the calibration map monotonic. This added flexibility is also visible in the fact that the beta calibration maps have three parameters (one for location, two for shape) as opposed to the logistic maps which have only two (one for location and one shape parameter fixing the slope at the midpoint). If the full flexibility is not required we can force the two shape parameters of the beta calibration family to be the same, which gives symmetric curves but still includes inverse sigmoids as well as sigmoids.

Our second contribution is that we connect beta calibration back to logistic calibration, by formulating it as a logistic regression problem over features constructed from the classifier's score s. In particular, the two-parameter version of beta calibration can be fitted by performing univariate logistic regression over the feature $\ln(s/(1-s))$, and the full three-parameter version can be fitted by means of bivariate logistic regression with features $\ln s$ and $-\ln(1-s)$. The two-parameter version of beta calibration with a and c where $a = b$ has been considered before as a linear-in-log-odds (LLO) calibration method (Lichtenstein, Fischhoff and Phillips, 1977; Turner et al., 2014) but without a justification.

We performed extensive experiments on 41 binary datasets, with Naive Bayes, Adaboost, Logistic Regression, SVM, Random Forest and MLP as base classifiers and evaluating both log-loss and Brier score. The experiments show that all versions of beta calibration outperform logistic calibration. We have also observed that beta calibration is a good alternative to isotonic calibration on smaller datasets, where isotonic calibration might overfit.

If the original classifier is already calibrated, then beta calibration learns a function close to the identity. On this we have built a statistical test to recognise if the model deviates from being well-calibrated. We have shown that a low p-value from this test suggests that applying beta calibration would lead to the reduction in losses.

There are several avenues for future work. As beta calibration is directly minimising log-loss it is perhaps no surprise that it outperforms isotonic calibration for log-loss, but this situation is reversed (although not significantly) for Brier score, and we plan to obtain a better understanding of this. It would also be interesting to formulate minimising Brier score as an alternative optimisation task.

Acknowledgements

This work was supported by the SPHERE Interdisciplinary Research Collaboration, funded by the UK Engineering and Physical Sciences Research Council under grant EP/K031910/1. MK was supported by the Estonian Research Council under grant PUT1458. TSF was financially supported by CNPq (Brazilian National Council for Scientific and Technological Development). All data used are openly available from the UCI repository (Lichman, 2013).

References

Cohen, I. and Goldszmidt, M. (2004). Properties and benefits of calibrated classifiers. In *European Conference on Principles of Data Mining and Knowledge Discovery* 125–136. Springer.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Machine Learning Research* **7** 1–30. MR2274360

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

Ferri, C., Flach, P. and Hernández-Orallo, J. (2003). Improving the AUC of Probabilistic Estimation Trees. In *14th Eur. Conf. on Machine Learning, (ECML'03)* 121–132. Springer.

Friedman, J., Hastie, T. and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics* **28** 337–407. MR1790002

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.* **11** 10–18.

Jones, E., Oliphant, T., Peterson, P. et al. (2001). SciPy: Open source scientific tools for Python.

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

Kull, M. and Flach, P. (2015). Novel Decompositions of Proper Scoring Rules for Classification: Score Adjustment as Precursor to Calibration. In *Machine Learning and Knowledge Discovery in Databases (ECML-PKDD'15)* 68–85. Springer.

Lichman, M. (2013). UCI Machine Learning Repository.

Lichtenstein, S., Fischhoff, B. and Phillips, L. D. (1977). Calibration of probabilities: The state of the art. In *Decision making and change in human affairs* 275–324. Springer.

Niculescu-Mizil, A. and Caruana, R. (2005). Predicting Good Probabilities with Supervised Learning. In *Proc. 22nd Int. Conf. on Machine Learning (ICML'05)* 625–632.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *J. Machine Learning Research* **12** 2825–2830.

Platt, J. (2000). Probabilities for SV machines. In *Advances in Large Margin Classifiers* (A. Smola, P. Bartlett, B. Schölkopf and D. Schuurmans, eds.) 61–74. MIT Press.

Provost, F. and Fawcett, T. (2001). Robust classification for imprecise environments. *Machine learning* **42** 203–231.

Turner, B. M., Steyvers, M., Merkle, E. C., Budescu, D. V. and Wallsten, T. S. (2014). Forecast aggregation via recalibration. *Machine Learning* **95** 261–289. MR3206174

Zadrozny, B. and Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates In *Proc. 8th Int. Conf. on Knowledge Discovery and Data Mining (KDD'02)* 694–699. ACM.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009). Multi-class AdaBoost. *Statistics and its Interface* **2** 349–360.