Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers

Meelis Kull
University of Bristol
University of Tartu

Telmo de Menezes e Silva Filho
Universidade Federal de Pernambuco
Centro de Informática

Peter Flach
University of Bristol

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 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 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 Naive Bayes and Adaboost.

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. 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 averaged over all possible true classes. The better our estimates of the class posterior are, the closer we get to the (irreducible) Bayes risk. A sufficiently calibrated classifier can be simply thresholded at a threshold directly derived from the misclassification costs. 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.

Some learning algorithms are designed to yield well-calibrated probabilities. These include decision trees, whose leaf probabilities are optimal on the training set (Provost and Domingos, 2003); as trees suffer from high variance, using Laplace smoothing and no pruning is recommended (Ferri et al., 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 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 pair-adjacent-violators (Zadrozny and Elkan, 2002; Fawcett and Niculescu-Mizil, 2007).

Isotonic calibration is a non-parametric method that 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 corre-

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

sponding bin can be derived. Hence the resulting calibration map is a non-decreasing, piecewise constant function. *Logistic calibration* is a parametric method which assumes that the scores within each class are normally distributed with the same variance, from which the familiar sigmoidal calibration map can be derived with 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. Being a parametric model, logistic calibration tends to require less labelled data to fit the sigmoid; however, it can produce bad results due to model mismatch.

The main contributions of this paper are (i) a demonstration that such model mismatch is a real danger 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 concludes.

2 LOGISTIC CALIBRATION

2.1 What Is 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. 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 $s_i = f(x_i)$ 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, where 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):

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

For any fixed model f there exists a uniquely determined calibration map which produces perfectly calibrated probabilities on the given dataset. That calibration map can be defined as $\mu(s_i) = \mathbb{E}[Y|f(X) = s_i]$. 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 different score on each training instance, then the training-perfect calibration map would produce only the 0/1-probabilities $\mu(s_i) = y_i$, which in most cases would be overly confident and overfitting.

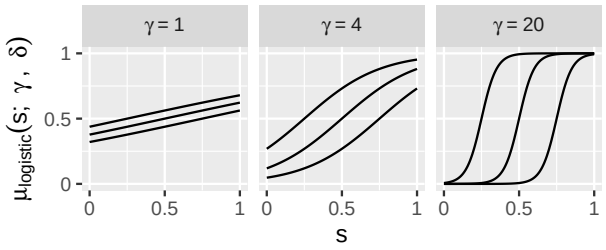


Figure 1: Examples of logistic curves with parameters $\gamma \in \{1, 4, 20\}$, $m \in \{0.25, 0.5, 0.75\}$ and $\delta = -m\gamma$.

2.2 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:

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

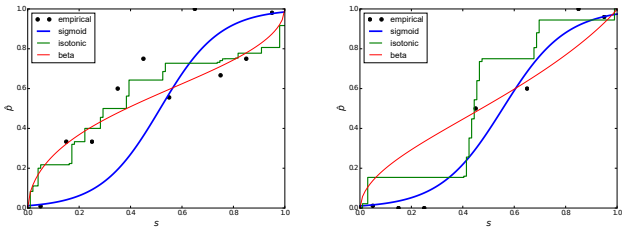
where γ, δ are real-valued parameters with $\gamma \geq 0$ to ensure that the calibration map is monotonically non-decreasing. Monotonicity is enforced assuming that higher model output scores suggest higher probability to be positive. For easier interpretability we introduce the parameter $m = -\delta/\gamma$.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

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

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 1 shows a variety of shapes that the logistic calibration map can take.

2.3 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. I.e., we need to measure how good the probability estimates $\hat{p}_i = \mu_{\text{logistic}}(s_i; \gamma, \delta)$ are, given the actual labels y_i. A well-known method to evaluate any 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 $-\ln(1 - \hat{p}_i)$. E.g., the fully confident predictions $\hat{p}_i = 0$ and $\hat{p}_i = 1$ incur loss 0 if correct, and loss ∞ if wrong,



(a) landsat-satellite (b) vowel

Figure 2: Calibration maps for two datasets. The x-axis represents uncalibrated scores. The "empirical" dots show the true-positive rates obtained from 10 bins of the uncalibrated scores produced by Adaboost.

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 $\hat{\mathbf{p}} = (\hat{p}_1, \ldots, \hat{p}_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$. 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, such as WEKA (Hall et al., 2009), Scikit-learn (Pedregosa et al., 2011), etc.

2.4 Why Logistic Calibration Can Fail

As mentioned in the introduction, logistic calibration can fail if its parametric assumptions are not met. We now demonstrate that this failure can be substantial and can actually lead to 'calibrated' scores that are worse than the original. Figure 2 shows two datasets where the score distortions (black dots) are clearly not sigmoidal. Isotonic calibration (green line) captures this but logistic calibration (blue) line results in a very poor fit, particularly in the left figure. The red line shows that our proposed beta calibration method provides a similar fit to isotonic calibration on this dataset. Note that this calibration map has an inverse-sigmoid shape, which is outside of the logistic family of calibration maps. This is no coincidence: the inverse sigmoid is appropriate for classifiers that tend to produce extreme scores close to 0 or 1, such as Adaboost. The logistic family, on the other hand, assumes that scores are too close to the midpoint and need to be pulled to the extremes.

On the right we see a dataset again without a clear sigmoidal pattern in the uncalibrated scores. Here isotonic calibration provides the best fit, but beta calibration learns a calibration map that is almost the identity, which at least does not make matters worse like logistic calibration does. Note that the identity map is not a member of the logistic family.

In order to derive beta calibration from first principles we first revisit a derivation of logistic calibration itself.

2.5 Logistic Calibration from First Principles

We 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]

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

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 with unit variance and with the means $s^+ = m + \gamma/2$ and $s^- = m - \gamma/2$ on positives and negatives, respectively.

[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 $1/2$.

3 BETA CALIBRATION

3.1 Beta Calibration from First Principles

The derivation of logistic calibration assumes normal distribution of scores within each class. For probabilistic classifiers such as Naive Bayes the Gaussians are unreasonable due to infinite support, because the probabilities are always in the range $[0,1]$. Hence it makes sense to derive an alternative parametric family of calibration functions using distributions with finite support. A natural choice 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.

Now 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

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

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 use the parametrisation with $K = e^{-c}$:

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

As before, we can 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 two beta distributions for the scores on positives and negatives. For 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}.

3.2 Examples of Beta Calibration

A simple 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 perfectly calibrated feature with values $\mathbf{x} = (x_1,\ldots,x_n)$ on the training instances. Naive Bayes would consider these features independent and output $\mathbf{s} = \frac{\mathbf{x}^k}{\mathbf{x}^k+(1-\mathbf{x})^k}$ on these instances. This pushes the probability estimates towards the extremes and the perfect calibration map must bring these back to their original calibrated values $\mathbf{x}$. Since $\mathbf{s}/(1-\mathbf{s}) = (\mathbf{x}/(1-\mathbf{x}))^k$, the perfect calibration map is $\mathbf{x} = 1/(1+\left(\frac{\mathbf{s}^{1/k}}{(1-\mathbf{s})^{1/k}}\right)^{-1})$, which belongs to the beta calibration family with the parameters $a = b = 1/k$ and $c = 0$.

As another example, suppose that we do not know that the scores $\mathbf{s}$ are already calibrated and we still apply 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 the 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 shows cases where $a = b$; we refer to this as beta[a=b] calibration. On the bottom right we see the familiar sigmoid 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. On the top left we see pure inverse-sigmoid 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.

The off-diagonal panels 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.

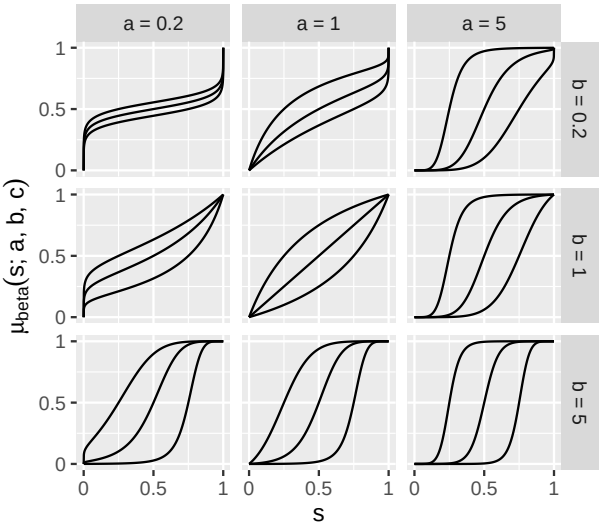


Figure 3: Examples of beta curves with parameters $a,b \in \{0.2,1,5\}$, $m \in \{0.25,0.5,0.75\}$ and $c = b\ln(1-m) - a\ln m$.

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.

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.

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

□

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 there 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) for fitting the beta[a=b] calibration maps, as shown in Algorithm 1. Note that the operators in the algorithm apply on vectors component-wise.

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** $\hat{\mathbf{p}}_{\mathsf{test}}$

Furthermore, it turns out that 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 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.

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

□

One subtle issue with Algorithms 1 and 2 is that in principle fitting might result in either $a<0$ or $b<0$ or both, yielding calibration maps that are either monotonically decreasing

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

or not monotonic at all. This did not ever happen in our experiments but if this situation is important to be avoided 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.

4 EXPERIMENTS

We evaluated the effect of applying beta calibration and its variations to the scores produced by Naive Bayes and Adaboost on 41 datasets from UCI (Lichman, 2013), see Table 1 for details. Multiclass datasets were transformed into binary by calling the biggest class positive and the other classes 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. Supplementary material additionally evaluates accuracy.

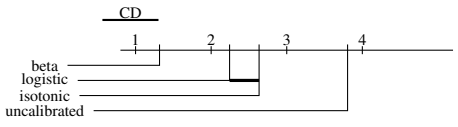
The results were obtained from 10 times 5-fold cross-validation, totalling 50 executions. Within each execution we used 3-fold internal cross-validation with 2 folds for learning the model and 1 for fitting the calibration map. Thus, three calibrated classifiers were generated during each execution, 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 (Platt, 2000). All experiments were written in Python and the code is publicly available[2]. The detailed tables of results are available in the Supplementary material.

For Naive Bayes (NB), we used the implementation provided by Scikit-learn (Pedregosa et al., 2011). For boosting we used 200 decision stumps as weak learners and implemented two different versions of the standard Adaboost algorithm. The first is the original Adaboost with probabilities extracted in the standard way as in (Friedman et al., 2000), we refer to it as Ada-O. The second is the one implemented in Scikit-learn's based on Adaboost method SAMME (Zhu et al., 2009), we refer to it as Ada-S. These versions turn out to be quite different, as Ada-O tends to push the probabilities to the extremes similarly to NB, whereas Ada-S pulls them towards 0.5. Therefore, we expect beta calibration to outperform logistic calibration on Ada-O and NB, while being comparably good on Ada-S.

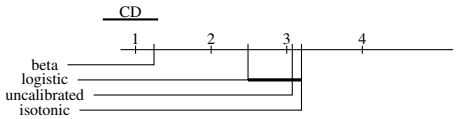
For statistical comparisons between the calibration methods we followed (Demšar, 2006) and conducted the Friedman test based on the average ranks across the data sets to verify whether the differences between algorithms were statistically significant. In case of significance at 5% confidence level we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

We first compared the full 3-parameter beta calibration with logistic and isotonic calibration methods, as well as with the uncalibrated probabilities, across all 3x2 settings (NB, Ada-S, Ada-O; LL, BS). The critical difference diagrams are shown in Figure 4 for LL and in Figure 5 for BS. As expected, beta calibration was significantly better than logistic calibration on NB and Ada-O, both for LL and BS. Furthermore, on LL beta calibration was even significantly better than isotonic calibration, while being on par under

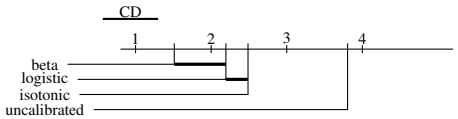
[2] https://betacal.github.io



(a) Naive Bayes

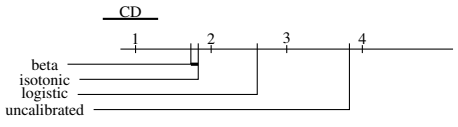


(b) Ada-O

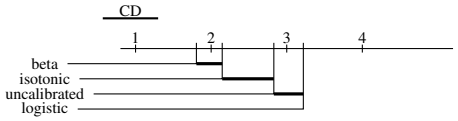


(c) Ada-S

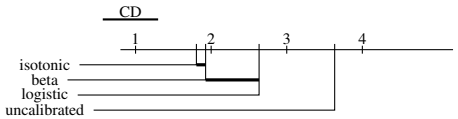
Figure 4: Critical difference diagrams for log-loss with Naive Bayes, Ada-O and Ada-S as base classifiers, Friedman test p-values are $6.9e{-}17$, $1.0e{-}12$ and $4.7e{-}15$, respectively.



(a) Naive Bayes



(b) Ada-O

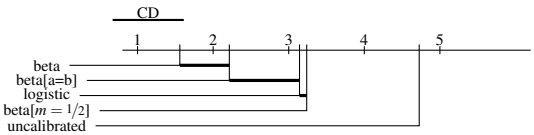


(c) Ada-S

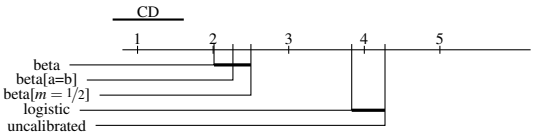
Figure 5: Critical difference diagrams for Brier score with Naive Bayes, Ada-O and Ada-S as base classifiers, Friedman test p-values are $1.0e{-}14$, $3.7e{-}06$ and $5.8e{-}13$, respectively.

BS. With Ada-S 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 and Ada-O according to LL. Full results on Ada-O for LL are shown in Table 2.

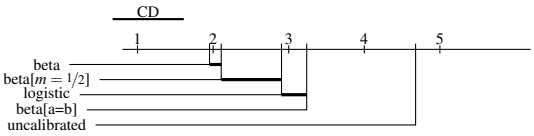
We then continued to compare the variants of beta calibration, leaving the non-parametric isotonic out of the picture. The critical difference diagrams for this analysis are shown



(a) Naive Bayes

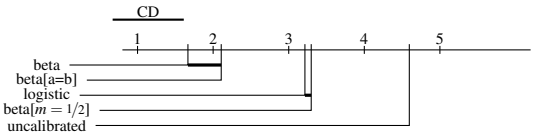


(b) Ada-O

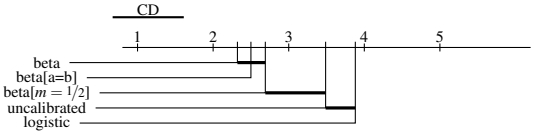


(c) Ada-S

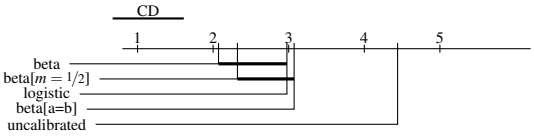
Figure 6: Critical difference diagrams for log-loss on parametric methods with Naive Bayes, Ada-O and Ada-S classifiers, Friedman test p-value are $1.2e{-}20$, $1.6e{-}15$ and $2.8e{-}17$.



(a) Naive Bayes



(b) Ada-O



(c) Ada-S

Figure 7: Critical difference diagrams for Brier score on parametric methods with Naive Bayes, Ada-O and Ada-S classifiers, Friedman test p-values are $2.4e{-}18$, $1.4e{-}06$ and $1.6e{-}13$.

in Figure 6 for LL and in Figure 7 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 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 regression, especially if the model has pushed the scores towards the extremes.

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

Beta[a=b] calibration has comparable running time to logistic calibration and the 3-parameter version can be slightly slower. However, both calibration methods are usually orders of magnitude faster than learning of 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.

5 CONCLUDING REMARKS

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 et al., 1977; Turner et al., 2014) but without a justification.

We performed extensive experiments on 41 binary datasets, with Naive Bayes and Adaboost 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.

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 get 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. 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

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

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

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

J. Friedman, T. Hastie, R. Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, 28(2):337–407, 2000.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009. ISSN 1931-0145.

E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001. URL `http://www.scipy.org/`.

M. Kull and P. Flach. 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)*, pages 68–85. Springer, 2015.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

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

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

J. Platt. Probabilities for SV machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Adv. Large Margin Classifiers*, pages 61–74. MIT Press, 2000.

F. Provost and P. Domingos. Tree induction for probability-based ranking. *Machine Learning*, 52(3):199–215, 2003.

B. M. Turner, M. Steyvers, E. C. Merkle, D. V. Budescu, and T. S. Wallsten. Forecast aggregation via recalibration. *Machine Learning*, 95(3):261–289, 2014.

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

J. Zhu, H. Zou, S. Rosset, and T. Hastie. Multi-class AdaBoost. *Statistics and its Interface*, 2(3):349–360, 2009.