
Peer reviewed version

Link to publication record in Explore Bristol Research
PDF-document

This is the author accepted manuscript (AAM). The final published version (version of record) is available online via PMLR at http://proceedings.mlr.press/v54/kull17a.html. 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
Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers

Meelis Kull
University of Bristol
Telmo de Menezes e Silva Filho
Universidade Federal de Pernambuco
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-
Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers

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 \( \mu \) 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 \( x_1, \ldots, x_n \) 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.

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:

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

where \( \gamma, \delta \) 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 \( m = \gamma / \delta \). This implies \( \delta = m \gamma \), yielding the following alternative parametrisation:

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

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 \( \infty \) 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:

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

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

\[
LL(\hat{p}, y) = -\ln \prod_{i=1}^{n} \hat{p}_i^{y_i} (1 - \hat{p}_i)^{1-y_i} = -\ln \left( \prod_{y_i=1} \hat{p}_i \prod_{y_i=0} (1 - \hat{p}_i) \right)
\]

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 \( s \) and label \( y \), 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-sigmoidal 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 \( \sigma^2 \). This gives class-specific probability density functions (PDFs)

\[
p(s+) = C \exp[-(s-s^+)^2/(2\sigma^2)]
\]

\[
p(s-) = C \exp[-(s-s^-)^2/(2\sigma^2)]
\]

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

\[
LR(s) = \frac{p(s+)}{p(s-)} = \exp[-(s-s^+)^2 + (s-s^-)^2]/(2\sigma^2)]
\]

\[
= \exp[(2(s^+-s^-)s-(s^2+s^-^2))/(2\sigma^2)]
\]

\[
= \exp[(s^+-s^-)/\sigma^2(s-(s^++s^-)/2)]
\]

\[
= \exp[\gamma(s-m)]
\]

where \( \gamma = (s^+-s^-)/\sigma^2 \) 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:

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

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.

---

This text has been edited for clarity and conciseness, focusing on the key aspects of the logistic calibration and the conditions under which it fails or succeeds. The mathematical expressions and diagrams are kept as close as possible to the original while ensuring readability and coherence.
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 $\alpha_0, \beta_0$ and $\alpha_1, \beta_1$, respectively. The likelihood ratio then becomes

$$LR(s; \alpha_0, \beta_0, \alpha_1, \beta_1) = \frac{s^{\alpha_0-1}(1-s)^{\beta_0-1}}{B(\alpha_0, \beta_0)} / \frac{s^{\alpha_1-1}(1-s)^{\beta_1-1}}{B(\alpha_1, \beta_1)}$$

$$= \frac{s^{\alpha_0-\alpha_1}(1-s)^{\beta_0-\beta_1}}{B(\alpha_0, \beta_0)} / B(\alpha_1, \beta_1)$$

$$= \frac{s^a}{(1-s)^b} / K$$

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 $K = e^{-c}$:

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

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

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

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 $B(x+a, y)/B(x, x+y)$ is 0 for $x = 1, y = 0$ but tends to $\infty$ 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 $x = (x_1, \ldots, x_k)$ on the training instances. Naive Bayes would consider these features independent and output $s = \frac{x}{s+1-(1-x)}$ 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 $x$. Since $s/(1-s) = (x/(1-x))^k$, the perfect calibration map is $x = 1/(1 + \left(s^k/(1-s)^k \right)^{-1})$, 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 $s$ 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

$$LR(s; a, b, c) = LR(s; a, b, c/\ln(1-m) - a \ln m)$$

$$= \frac{s^a}{(1-s)^b} / \frac{m^a}{(1-m)^b}$$

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.
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 $c, s \in \mathbb{R}$: $\mu_{\beta}(s; a, a, c) = \mu_{\text{logistic}}(\ln \frac{s}{1-s}; a, c)$.

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

$$LR_{\text{logistic}}(\ln \frac{s}{1-s}; a, c) = \exp[a \ln s - \ln(1-s) + c]$$

$$= \left(\frac{s}{1-s}\right)^a e^c = LR_{\beta}(s; a, a, c)$$

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.

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 $c, s \in \mathbb{R}$: $\mu_{\beta}(s; a, b, c) = \mu_{\text{bilogistic}}(\ln s, -\ln(1-s); a, b, c)$, where $\mu_{\text{bilogistic}}(s', s''; a, b, c) = 1/(1 + 1/\exp[as' + bs'' + c])$ is the bivariate logistic regression model family.

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

$$LR_{\text{bilogistic}}(\ln s, -\ln(1-s); a, b, c)$$

$$= \exp[a \ln s - b \ln(1-s) + c]$$

$$= \frac{s^a}{(1-s)^b} e^c = LR_{\beta}(s; a, b, c)$$

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

![Figure 3: Examples of beta curves with parameters $a, b \in \{0.2, 1, 5\}$, $m \in \{0.25, 0.5, 0.75\}$ and $c = b \ln(1 - m) - a \ln m$.](image)
Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers

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

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

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 logloss (LL), which are both proper scoring rules (Kull and Flach 2013) 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 for 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 (Deˇsmaˇr 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
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 in Figure 4 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.
fier itself. Therefore, the overall time for learning a clas
slightly slower. However, both calibration methods are usu-
logistic calibration and the 3-parameter version can be
Beta[a=b] calibration has comparable running time to
logistic calibration and the 3-parameter version can be
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 clas-
sifier each follow a beta distribution. This is particularly
suitable for classifiers that score on a bounded scale, for
which the Gaussian assumption underlying logistic calibra-
tion 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 homoscedas-
ticity to keep the calibration map monotonic. This added
flexibility is also visible in the fact that the beta calibra-
tion 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 bi-
variate 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-
hoods (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 logis-
tic calibration. We have also observed that beta calibra-
tion is a good alternative to isotonic calibration on smaller
datasets, where isotonic calibration might overfit.

There are several avenues for future work. As beta calibra-
tion 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 Interdisci-
plinary Research Collaboration, funded by the UK En-
gineering and Physical Sciences Research Council under
grant EP/K031910/1. TSF was financially supported by
CNPq (Brazilian National Council for Scientific and Tech-
nological Development). All data used are openly available
from the UCI repository [Lichman, 2013].

Beta[a=b] calibration has comparable running time to
logistic calibration and the 3-parameter version can be
略 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.

**5 CONCLUDING REMARKS**

We introduced a rich and flexible family of calibration
maps that includes both sigmoids and inverse sigmoids, as

## 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).

<table>
<thead>
<tr>
<th>dataset</th>
<th>uncalibrated</th>
<th>beta</th>
<th>isotonic</th>
<th>logistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>0.6124</td>
<td>0.6121</td>
<td>0.6264</td>
<td>0.6141</td>
</tr>
<tr>
<td>autos</td>
<td>0.4274</td>
<td>0.2832</td>
<td>0.4161</td>
<td>0.2872</td>
</tr>
<tr>
<td>balance</td>
<td>0.0494</td>
<td>0.0370</td>
<td>0.0413</td>
<td>0.0382</td>
</tr>
<tr>
<td>car</td>
<td>0.1142</td>
<td>0.1097</td>
<td>0.1224</td>
<td>0.1191</td>
</tr>
<tr>
<td>cleva</td>
<td>0.4967</td>
<td>0.4171</td>
<td>0.5234</td>
<td>0.4292</td>
</tr>
<tr>
<td>credit</td>
<td>0.3603</td>
<td>0.3412</td>
<td>0.4444</td>
<td>0.3381</td>
</tr>
<tr>
<td>dermato</td>
<td>0.0364</td>
<td>0.0192</td>
<td>0.0353</td>
<td>0.0211</td>
</tr>
<tr>
<td>diabete</td>
<td>0.5063</td>
<td>0.4841</td>
<td>0.5254</td>
<td>0.4952</td>
</tr>
<tr>
<td>ecoli</td>
<td>0.3264</td>
<td>0.1452</td>
<td>0.2243</td>
<td>0.1592</td>
</tr>
<tr>
<td>flare</td>
<td>0.4041</td>
<td>0.4042</td>
<td>0.4214</td>
<td>0.4083</td>
</tr>
<tr>
<td>german</td>
<td>0.5113</td>
<td>0.5062</td>
<td>0.5384</td>
<td>0.5061</td>
</tr>
<tr>
<td>glass</td>
<td>0.6964</td>
<td>0.4892</td>
<td>0.5603</td>
<td>0.4851</td>
</tr>
<tr>
<td>heart-s</td>
<td>0.5704</td>
<td>0.4271</td>
<td>0.5573</td>
<td>0.4432</td>
</tr>
<tr>
<td>hepatitis</td>
<td>0.8054</td>
<td>0.3920</td>
<td>0.4313</td>
<td>0.4112</td>
</tr>
<tr>
<td>horse</td>
<td>0.6424</td>
<td>0.4132</td>
<td>0.5243</td>
<td>0.4185</td>
</tr>
<tr>
<td>ionosph</td>
<td>0.3814</td>
<td>0.2032</td>
<td>0.2963</td>
<td>0.2272</td>
</tr>
<tr>
<td>iris</td>
<td>0.1274</td>
<td>0.0002</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td>landsat</td>
<td>0.0412</td>
<td>0.0400</td>
<td>0.0433</td>
<td>0.0450</td>
</tr>
<tr>
<td>letter</td>
<td>0.0378</td>
<td>0.0352</td>
<td>0.0335</td>
<td>0.0363</td>
</tr>
<tr>
<td>libras</td>
<td>0.5894</td>
<td>0.1001</td>
<td>0.1843</td>
<td>0.1122</td>
</tr>
<tr>
<td>lung-ca</td>
<td>0.1934</td>
<td>0.1394</td>
<td>0.1893</td>
<td>0.1392</td>
</tr>
<tr>
<td>mfeat-k</td>
<td>0.0624</td>
<td>0.0271</td>
<td>0.0483</td>
<td>0.0326</td>
</tr>
<tr>
<td>mfeat-m</td>
<td>0.0404</td>
<td>0.0142</td>
<td>0.0261</td>
<td>0.0142</td>
</tr>
<tr>
<td>mfeat-z</td>
<td>0.0954</td>
<td>0.0326</td>
<td>0.0639</td>
<td>0.0393</td>
</tr>
<tr>
<td>mushroom</td>
<td>0.0004</td>
<td>0.0003</td>
<td>0.0002</td>
<td>0.0000</td>
</tr>
<tr>
<td>optdigi</td>
<td>0.0352</td>
<td>0.0331</td>
<td>0.0444</td>
<td>0.0403</td>
</tr>
<tr>
<td>page-bl</td>
<td>0.0932</td>
<td>0.0899</td>
<td>0.0983</td>
<td>0.1092</td>
</tr>
<tr>
<td>pendigi</td>
<td>0.0172</td>
<td>0.0172</td>
<td>0.0234</td>
<td>0.0213</td>
</tr>
<tr>
<td>scene-c</td>
<td>0.3844</td>
<td>0.3646</td>
<td>0.3762</td>
<td>0.3785</td>
</tr>
<tr>
<td>segment</td>
<td>0.0203</td>
<td>0.0102</td>
<td>0.0234</td>
<td>0.0132</td>
</tr>
<tr>
<td>shuttle</td>
<td>0.0002</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td>sonar</td>
<td>0.9414</td>
<td>0.4041</td>
<td>0.4863</td>
<td>0.4404</td>
</tr>
<tr>
<td>spambas</td>
<td>0.1622</td>
<td>0.1622</td>
<td>0.1734</td>
<td>0.1673</td>
</tr>
<tr>
<td>tic-tac</td>
<td>0.3794</td>
<td>0.3333</td>
<td>0.3403</td>
<td>0.3392</td>
</tr>
<tr>
<td>vehicle</td>
<td>0.0772</td>
<td>0.0681</td>
<td>0.1314</td>
<td>0.0773</td>
</tr>
<tr>
<td>vowel</td>
<td>0.0764</td>
<td>0.0711</td>
<td>0.0949</td>
<td>0.0885</td>
</tr>
<tr>
<td>waveform</td>
<td>0.2534</td>
<td>0.2522</td>
<td>0.2643</td>
<td>0.2714</td>
</tr>
<tr>
<td>wdbc</td>
<td>0.2564</td>
<td>0.0899</td>
<td>0.1413</td>
<td>0.1072</td>
</tr>
<tr>
<td>wpbc</td>
<td>1.0164</td>
<td>0.5002</td>
<td>0.4961</td>
<td>0.5033</td>
</tr>
<tr>
<td>yeast</td>
<td>0.5102</td>
<td>0.5091</td>
<td>0.5434</td>
<td>0.5143</td>
</tr>
<tr>
<td>zoo</td>
<td>0.1324</td>
<td>0.0133</td>
<td>0.0131</td>
<td>0.0132</td>
</tr>
</tbody>
</table>

**rank** 3.20 1.27 3.12 2.41
References


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.