A Practical Method Based on Bayes Boundary-Ness for Optimal Classifier Parameter Status Selection

David Ha¹ · Yuya Tomotoshi¹ · Masahiro Senda¹ · Hideyuki Watanabe² · Shigeru Katagiri¹ · Miho Ohsaki¹

Received: 11 February 2019 / Revised: 25 March 2019 / Accepted: 29 March 2019 / Published online: 11 May 2019
© The Author(s) 2019

Abstract

We propose a novel practical method for finding the *optimal* classifier parameter status corresponding to the Bayes error (minimum classification error probability) through the evaluation of estimated class boundaries from the perspective of *Bayes boundary-ness*. While traditional methods approach classifier optimality from the angle of minimization of the estimated classification error probabilities, we approach it from the angle of optimality of the estimated classification boundaries. The optimal classification boundary consists solely of *uncertain samples*, whose posterior probability is equal for the two classes separated by the boundary. We refer to this essential characteristic of the boundary as “Bayes boundary-ness”, and use it to measure how optimal the estimated boundary is. Our proposed method achieves the optimal parameter status using the training data only once, in contrast to such traditional methods as Cross-Validation (CV), which demand separate validation data and often require a number of repetitions of training and validation. Moreover, it can be directly applied to any type of classifier, and potentially to any type of sample. In this paper, we first elaborate on our proposed method that implements the Bayes boundary-ness with an entropy-based *uncertainty measure*. Next, we analyze the mathematical characteristics of the uncertainty measure adopted. Finally, we evaluate the method through a systematic experimental comparison with CV-based Bayes boundary estimation, which is known to be highly reliable in the Bayes error estimation. From the analysis, we rigorously show the theoretical validity of our adopted uncertainty measure. Moreover, from the experiment, we successfully demonstrate that our method can closely approximate the CV-based Bayes boundary estimate and its corresponding classifier parameter status with only a single-shot training over the data in hand.

Keywords Pattern recognition · Classification · Classifier model selection · Class boundary uncertainty

1 Introduction

In the statistical approach to the development of pattern classifiers, the ultimate goal of classifier training is to find the *optimal* classifier parameter (class model parameters) status that leads to the minimum classification error probability (also called Bayes error). As a result, many classifier training methods have been vigorously investigated to achieve this goal through accurate estimation

of the classification error probability (e.g. [1–3]). However, the error probability requires an infinite number of samples. Therefore, its accurate estimation is inherently difficult in real-world situations, where only a finite number of samples are available [4].

Simple but standard approaches for solving the above difficulty include resampling methods such as Hold-Out (HO), Cross-Validation (CV), Leave-One-Out-CV (LOO-CV) [5], and Bootstrap [6]. HO splits a given sample set into a training sample subset and a validation sample subset, and then estimates the error probability (or classifier status) by using the pair of these two subsets. This raises the issue of how given samples should be split, which inevitably decreases either the training or the validation samples, and therefore degrades the error probability estimation. In contrast, CV, LOO-CV, and Bootstrap give an accurate and reliable estimate of the error probability conditioned by the number of resampling repetitions. CV solves the above degradation issue by producing multiple pairs of subsets

✉ David Ha
euq3101@mail4.doshisha.ac.jp

Shigeru Katagiri
skatagir@mail.doshisha.ac.jp

¹ Doshisha University, Kyoto, Japan

² Advanced Telecommunications Research Institute International, Kyoto, Japan

with a certain splitting ratio, repeating the estimation for each pair, and then averaging the estimates over the pairs; LOO-CV increases the reliability of CV by using as many subset pairs as possible. Bootstrap produces many sample sets by applying sampling-with-replacement to the given set, and increases the estimation reliability by repeating the estimation on each resampled set, while averaging the estimates over those resampled sets. However, such repetitions can be prohibitively time-consuming and unsuitable for real-life trainings that use larger number of samples, large-scale classifiers, or longer classifier training (e.g. [7]).

In contrast to the resampling methods, Structural Risk Minimization (SRM) [8] provides an upper bound on the error probability using only training data; it is also applicable to any classifier in principle. However, its application to classifier parameter status selection is not straightforward. Indeed, the derivation of the upper bound for a given classifier is usually difficult [4, 9], and the upper bound itself is known to be loose in practice [4].

Information Criteria (IC) such as the Akaike Information Criterion (AIC) [10] and the Bayes Information Criterion (BIC) [11] also require only training data. However, they often impose assumptions on classifier models that do not necessarily hold in practice [12]; moreover, they are basically designed for modeling sample distribution. Classification tasks do not focus on accurately modeling sample distributions, but on modeling the boundary between classes. Therefore, IC are not best suited for the estimation of the classification error probability or of the classifier parameter status [13–15]. Moreover, IC are able to evaluate a selected number of parameters (selected model size), but they are not fully suited to the evaluation of trained parameter statuses (parameter values).

Among the recent discriminative training methods, Minimum Classification Error (MCE) training [16, 17] pursues the minimum classification error probability in a direct manner by virtually increasing the training samples and minimizing the classification error counts. However, the effectiveness of the virtual samples in bridging the gap between a practical finite-sample situation and the ideal infinite-sample condition is not yet optimal [18].

Motivated by the above limitations of existing methods, we explored a new method for finding the optimal classifier parameter status. Our new method: 1) provides optimal (exact) values for classifier parameters instead of bounds in contrast to SRM, 2) can be easily and directly applied to any type of classifier in contrast to the IC-based methods, and 3) avoids a prohibitively long repetition of training/validation in contrast to the resampling methods. To this end, we focused on the property of the optimal classifier's parameter status (parameter value). The optimal parameter status corresponds to the Bayes error, and draws a class boundary

in which the on-boundary samples have equal posterior probabilities in terms of two dominant classes [2]. We call such a boundary a *Bayes boundary*. For every point on the Bayes boundary, classification is uncertain; but for every point off the Bayes boundary, classification is certain. Therefore, the higher the classification uncertainty is, the higher the *Bayes boundary-ness*¹ is. Importantly, the classification uncertainty, and in turn the Bayes boundary-ness, holds for a practical finite number of samples as well as for an ideal infinite number of samples; in principle, this relation holds regardless of the number of samples. Moreover, the relation holds regardless of the dimensionality of (vector) samples and independently from classifier selection. Therefore, we assume that our general concept of Bayes boundary-ness can be used to develop an effective method for finding the optimal classifier parameter status: 1) a Bayes boundary-ness score directly represents optimality in classifier parameter values, 2) measuring the Bayes boundary-ness does not assume classifier types, and 3) in principle, measuring the Bayes boundary-ness can be done using a single set of training samples without training/validation repetitions.

Similarly to estimates of error probability (and its minimum, i.e., Bayes error), estimates of Bayes boundary-ness using a finite number of samples cannot completely avoid deterioration in estimation quality. However, in contrast to error probability estimation, whose deteriorated result is not straightforwardly linked to degradation in the quality of Bayes boundary estimation, a deteriorated estimate of Bayes boundary-ness directly degrades the Bayes boundary estimation. In other words, the estimation of Bayes error is inevitably affected by sample distribution and thus its quality is not linearly linked to the quality of Bayes boundary estimation, while the quality of Bayes boundary-ness estimation is itself equivalent to such quality for the Bayes boundary estimation.

Encouraged by the above considerations, we preliminarily implemented the Bayes-boundary-ness-based concept as a method adopting an entropy-based *uncertainty measure* and investigated its effectiveness in the task of finding the optimal size of a multi-prototype-based classifier, i.e. the number of prototypes per class for a small number of two-class classification problems [19]. Experimental results for the method showed its fundamental utility, but these results were not so helpful due to the insufficient implementation of procedures such as finding *near-boundary samples*, which are defined below, and calculating the entropy. We also applied our method to a task of optimally setting the width of a Gaussian kernel for Support Vector Machine (SVM), while making some improvements to the implementation

¹This term represents how close an estimated boundary is to the Bayes boundary.

[20, 21]. The experimental results [21] showed its utility more clearly than did the previous experiment [19], but there remained issues to be improved. For example, the improved implementation provided suboptimal classifier statuses for some difficult datasets; furthermore, even if the concept basically worked well, it was somewhat heuristically implemented, and thus its theoretical validity was not sufficiently supported.

Based on the above background, in this paper, we comprehensively introduce a new Bayes-boundary-ness-based method for selecting an optimal classifier parameter status. First, we elaborate on the concept and its implementation, where we use the entropy to measure the Bayes boundary-ness, i.e. classification uncertainty. Next, we mathematically analyze the procedure of estimating posterior probabilities for computing the uncertainty measure, and show its theoretical validity, namely its unbiased and small-variance nature in convergence as the number of training samples increases. Finally, we demonstrate its utility in comparison experiments with the most fundamental CV-based method, using SVM classifiers over one synthetic dataset and eleven real-life datasets.

In comparative experiments, the selection of competitors is an important issue. As cited above, only resampling methods like CV and Bootstrap can realistically compete with our proposed method in the sense of being classification-oriented and classifier model-free. On the other hand, especially for the SVM classifiers we adopt in our experiments, various attempts to find the optimal parameter status have been made, for example an approximation of LOO-CV-based Bayes error estimation [22], a minimum-description-length-like method for estimating the Bayes error [23], and an IC-based modeling for the Bayes error [24]. However, all these methods are specific to SVM, and they are not directly applicable to other types of classifiers. While [22] produces accurate results in small-sized datasets, it may be excessively time-consuming and inapplicable to large-sized datasets. Moreover, compared to [22], the other less time-consuming methods [23, 24] basically increase error in the Bayes error estimation, therefore they are basically not the most appropriate to estimate the Bayes boundary. Based on these previous results for SVM classifiers, and taking into account the fact that the previous studies treated LOO-CV or CV as the most reliable method for estimating the Bayes error, we adopted a CV-based method as our competitor in the comparative experiments (simply using a large test set to estimate the Bayes error would be the simplest competitor. However, for the real-life datasets, generating a large number of virtual samples is not necessarily appropriate because their sample distribution functions are unknown, and thus the CV-based method can be a reasonable selection for the competitor. For the synthetic dataset, we generated a large test set.). As shown in a later section, we carefully exam-

ined the necessary number of training/validation repetitions in CV to find accurate estimates of the Bayes boundary and its corresponding optimal classifier parameter status (via estimation of the Bayes error).

2 Classifier Training Problem

Given a pattern sample $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ is a D -dimensional pattern space, we consider a task of classifying $\mathbf{x}$ into one of J classes ($C_1, \dots, C_J$), based on the following classification decision rule:

$$C(\mathbf{x}) = C_k \text{ iff } k = \arg \max_j g_j(\mathbf{x}; \Lambda), \quad (1)$$

where $C(\cdot)$ is a classification operator, Λ is a set of classifier parameters, and $g_j(\mathbf{x}; \Lambda)$ is a discriminant function for C_j , which represents the degree of confidence to which the classifier assigns $\mathbf{x}$ to class C_j . A higher value for $g_j(\mathbf{x}; \Lambda)$ represents higher confidence.

The aim of classifier training is to find the parameter status Λ^* for which $\{g_j(\mathbf{x}; \Lambda^*)\}_{j \in [1, J]}$ draws the Bayes boundary B^* that corresponds to the ideal Bayes error (minimum classification error probability) condition.

At a given sample location $\mathbf{x}$, one dominant discriminant function score $g_j(\cdot)$ enables the classifier to assign $\mathbf{x}$ to class C_j . When the two highest scores are equal, the classifier cannot decide between the two corresponding classes, and we say that $\mathbf{x}$ lies on the estimated boundary between the two classes. The concept of boundary directly extends to three or more classes; however, equality among three or more scores at a sample location is less likely. Therefore, the rest of this paper basically assumes boundaries between two classes. We denote $B_{ij}(\Lambda)$ as the estimated boundary that separates C_j and C_i . Clearly, $B_{ij}(\Lambda)$ and $B_{ji}(\Lambda)$ are interchangeable.

3 Proposed Method for Two-Class Data

3.1 Outline

For simplicity of presentation, we first cover the case with only two classes, C_0 and C_1 . In this case, we abbreviate the estimated boundary $B_{01}(\Lambda)$ to $B(\Lambda)$. For this two-class problem, we aim to find as close an approximation as possible of the Bayes boundary among the estimated boundaries, each produced beforehand by classifier training.

Our procedure consists of two steps: Step 1 “near-boundary sample selection” and Step 2 “uncertainty measure computation” (Algorithm 1 and Fig. 1). Assume that some classifier parameter status Λ and its corresponding class boundary $B(\Lambda)$ are given from classifier training.

Then, we evaluate the similarity between $B(\Lambda)$ and B^* , which we denote as $U(\Lambda)$ and call the uncertainty measure. Using this, we find the status of Λ that corresponds to the highest $U(\Lambda)$ and provides the boundary closest to (ideally the same as) B^* .

Accurate computation of $U(\Lambda)$ requires on-boundary samples on $B(\Lambda)$. However, in practical situations where only a finite number of samples are accessible, we do not necessarily have on-boundary samples. Therefore, to compute $U(\Lambda)$, we have to use samples close to $B(\Lambda)$, which we call near-boundary samples and denote as $\mathcal{N}_B(\Lambda)$. “Samples on $B(\Lambda)$ ” is replaced by “samples in $\mathcal{N}_B(\Lambda)$ ” in practice. However, if no near-boundary set is found, then a default minimum classification uncertainty value of 0 is assigned to $U(\Lambda)$. This case is covered later in Section 4.2.

Algorithm 1 Proposed uncertainty-measure-based method for two-class data.

Input: Set of candidate statuses & set of training samples $\mathcal{T}$
Output: $\arg \max_{\Lambda} U(\Lambda)$

```

1 for  $\Lambda \in \text{set of candidate statuses}$  do
2   Find  $\mathcal{N}_B(\Lambda)$  (Step 1: 3.2)
3   If  $\mathcal{N}_B(\Lambda) \neq \emptyset$  then evaluate  $U(\Lambda)$  using  $\mathcal{N}_B(\Lambda)$  (Step 2: Section 3.3)
4   Else  $U(\Lambda) \leftarrow 0$ 
5 end
```

3.2 Step 1: Selection of Near-Boundary Samples $\mathcal{N}_B(\Lambda)$

Assume a set of training samples $\mathcal{T}$ (a in Fig. 1). Then, to obtain $\mathcal{N}_B(\Lambda)$, we apply the definition of being closest to $B(\Lambda)$ (Algorithm 2): first, generate *anchor* samples on $B(\Lambda)$, denoted by $\mathbf{x}_a$ (b in Fig. 1); next, select their nearest neighbors in $\mathcal{T}$, denoted by $\text{NN}(\mathbf{x}_a, \mathcal{T})$ (c in Fig. 1); finally, add $\text{NN}(\mathbf{x}_a, \mathcal{T})$ to $\mathcal{N}_B(\Lambda)$ (c and d in Fig. 1).

Searching the multi-dimensional pattern space for anchors can be costly. We can reduce this search to a single dimension based on the following observation (Case (A) in Fig. 2). Assume that a pair of training samples $\{\mathbf{x}, \mathbf{x}'\}$ are assigned different estimated class labels $\{\hat{C}_0, \hat{C}_1\}$ by a trained classifier, where $\hat{C}_j$ means a class index estimated by classification decision. On the segment connecting $\mathbf{x}$ and $\mathbf{x}'$, estimated boundary $B(\Lambda)$ passes through at least once, namely $\exists \alpha \in [0, 1] : \alpha \mathbf{x} + (1 - \alpha) \mathbf{x}' \in B(\Lambda)$, i.e. $\alpha \in [0, 1] : g_0(\alpha \mathbf{x} + (1 - \alpha) \mathbf{x}'; \Lambda) = g_1(\alpha \mathbf{x} + (1 - \alpha) \mathbf{x}'; \Lambda)$. For example, we can find α by dichotomy, which provides an interval $[a, b]$ as tight as possible and satisfies $\text{sign}(g_0(a; \Lambda) - g_1(a; \Lambda)) \neq \text{sign}(g_0(b; \Lambda) - g_1(b; \Lambda))$.

The selection process of a near-boundary sample can be stopped once the number of near-boundary samples n_{anc}^{curr}

does not show an increase from its previous value n_{anc}^{prev} , even after creating E more anchors or after a preset number L_M of random pairs has been used to create anchors. If all of the training samples have the same estimated labels, then the boundary is not reasonable, and we set a default minimum uncertainty measure score of 0.

Algorithm 2 Selection of $\mathcal{N}_B(\Lambda)$ for two-class data.

Input: Classifier parameters Λ trained on $\mathcal{T}$
Output: $\mathcal{N}_B(\Lambda)$

```

1  $l \leftarrow 0, \text{Stop} \leftarrow \text{False}, n_{anc}^{prev} \leftarrow 0, n_{anc}^{curr} \leftarrow 0$ 
2 if  $\forall \mathbf{x}, \mathbf{x}' \in \hat{C}_0$  or  $\forall \mathbf{x}, \mathbf{x}' \in \hat{C}_1$  then  $\mathcal{N}_B(\Lambda) \leftarrow \emptyset$ ;
3 else
4   while  $l < l_M$  and  $\text{Stop} = \text{False}$  do
5     Randomly choose  $\mathbf{x}, \mathbf{x}' \in \mathcal{T} : C(\mathbf{x}) \neq C(\mathbf{x}')$ 
6     Find  $\alpha_0 \in [0, 1] : g_0(\alpha_0 \mathbf{x} + (1 - \alpha_0) \mathbf{x}'; \Lambda) = g_1(\alpha_0 \mathbf{x} + (1 - \alpha_0) \mathbf{x}'; \Lambda)$ 
7     Find  $t_a = \text{NN}(\alpha_0 \mathbf{x} + (1 - \alpha_0) \mathbf{x}', \mathcal{T})$ 
8     If  $t_a \notin \mathcal{N}_B(\Lambda)$  then add to  $\mathcal{N}_B(\Lambda)$ 
9     if  $\text{loop} \% E = 0$  then
10       $n_{anc}^{curr} \leftarrow \text{card}(\mathcal{N}_B(\Lambda))$ ;
11      If  $n_{anc}^{curr} = n_{anc}^{prev}$  then  $\text{Stop} \leftarrow \text{True}$ ;
12       $n_{anc}^{prev} \leftarrow n_{anc}^{curr}$ 
13    end
14     $l \leftarrow l + 1$ 
15  end
16 end
17 return  $\mathcal{N}_B(\Lambda)$ 
```

3.3 Step 2: Computation of Uncertainty Measure $U(\Lambda)$

As for our uncertainty measure, we only require that it reach a maximum when posterior probabilities are equal and that it show a lower score when posterior probabilities are imbalanced. As one possible choice among others (e.g. Gini impurity [25]), we adopt Shannon entropy $H(\mathbf{x})$ to implement uncertainty measure $U(\Lambda)$ in the remainder of this paper. The local uncertainty at near-boundary sample $\mathbf{x}$ is therefore

$$H(\mathbf{x}) = - \sum_{j=0}^1 P(C_j | \mathbf{x}) \log(P(C_j | \mathbf{x})), \quad (2)$$

and the uncertainty measure of $B(\Lambda)$ is

$$U(\Lambda) = \frac{1}{N_B} \sum_{\mathbf{x} \in \mathcal{N}_B(\Lambda)} H(\mathbf{x}), \quad (3)$$

where N_B is the number of samples in $\mathcal{N}_B(\Lambda)$, i.e. near-boundary samples.

Figure 1 Graphical explanation of our two-step method. Samples with estimated class labels $\hat{C}_0$ and $\hat{C}_1$ are shown in green triangles and yellow squares, respectively. Generated anchors are shown in red dots.

Applying Eqs. 2 and 3 as such would require estimating the posterior probabilities for each near-boundary sample in $\mathcal{N}_B(\Lambda)$. Posterior probabilities can be simply estimated in an assumption-free way using the k -Nearest Neighbor (k NN) method. Given near-boundary sample $\mathbf{x}$, the posterior probability estimate is $\hat{P}(C_i|\mathbf{x}) = k_i/k$, where the number of C_i 's neighbors is k_i ($\sum_{i=1}^J k_i = k$). In this posterior estimation, we use only near-boundary samples in $\mathcal{N}_B(\Lambda)$ as candidates of k nearest neighbors. This restricted use can be seen as a filtering step to retain only samples relevant for the estimation of class posteriors on $B(\Lambda)$ (d in Fig. 1)).

A simple idea would be to apply the k NN posterior probability estimation to each near-boundary sample in $\mathcal{N}_B(\Lambda)$ to obtain the entropy in Eq. 2 and then apply (3). However, obtaining the optimal setting of the number of neighbors k for each $\mathbf{x}$ ($\in \mathcal{N}_B(\Lambda)$) could itself require a parameter selection procedure. Intuitively, we should locally

adapt k to the local sample density, since a higher density allows for a higher k . To implement this adaptive selection, we adopt a procedure that first obtains adaptive neighbors by iteratively and hierarchically using 2-means clustering and then next applies (3) to obtain local uncertainty measure scores (Algorithm 3).

A key concept of the adopted procedure is to compute the uncertainty measure at cluster centroids, each representing near-boundary samples as cluster members, instead of near-boundary samples; here, the cluster expresses the local sample density around its centroid (e and f in Fig. 1)). The procedure consists of two loops: an outer loop for R iterations with different initialization of clustering and an inner loop for running the hierarchical clustering for $\mathcal{N}_B(\Lambda)$.

At the r -th iteration of the outer loop ($r \in [1, R]$) of Algorithm 3, we first hierarchically apply the 2-means clustering to the near-boundary samples in $\mathcal{N}_B(\Lambda)$ until

Figure 2 Anchor generation based on a random pair $[a, b]$ belonging to a pair of different estimated classes $\hat{C}_0$ and $\hat{C}_1$, where we illustrate the need for a case-by-case procedure. In the two-class data case, segment ab strides only two class regions, while multiple boundaries can pass through the segment (case (A)). In the multi-class data

case, segment ab can stride multiple class regions and multiple boundaries can pass through the segment (case (B)); when the segment strides multiple class regions, we divide the segment in the class-by-class manner and set an anchor for every pair of two adjacent classes.

we can satisfy the following condition for clusters: in each cluster $p \in \mathcal{P}^{(r)}$, the number of its members $\text{card}(p)$ is equal to or larger than N_m and equal to or smaller than N_M (the inner loop of Algorithm 3). When executing 2-means clustering, we tested random initialization M times and then chose the initialization that provides the clusters with the smallest distortion rate. Here, $\text{card}(\cdot)$ represents cardinality, $\mathcal{P}^{(r)}$ is the set of clusters produced at the r -th iteration of the outer loop, and the minimum of cluster members N_m and the maximum of cluster members N_M are set simply to maintain reliability in the posterior probability estimation. Accordingly, we obtain multiple clusters or their corresponding cluster centroids. Next, at the centroid of each cluster p , we compute entropy $\sum_{j=0}^1 P(C_j|p) \log(P(C_j|p))$, or the local uncertainty measure, using the corresponding $\text{card}(p)$ cluster members as the neighbors of the k NN method. Finally, we obtain uncertainty measure score $\hat{U}^{(r)}(\Lambda)$ by averaging the entropy scores over all clusters in $\mathcal{P}^{(r)}$.

In the outer loop, we repeat the above inner loop of hierarchical clustering R times to further increase reliability in the uncertainty measure computation. Each run of the outer loop is controlled by the difference in random initialization for clustering. From the outer loop iteration, we finally obtain uncertainty measure score $\hat{U}(\Lambda)$, which is expected to appropriately represent the sample density around the class boundaries, by averaging $\hat{U}^{(r)}(\Lambda)$ ($r \in [1, R]$).

Algorithm 3 Computation of $U(\Lambda)$ using hierarchical clustering.

Input: $\mathcal{N}_B(\Lambda)$, N_m , N_M , M , R
Output: $\hat{U}(\Lambda)$

```

1 for  $r \in [1, R]$  do
2    $\mathcal{P}^{(r)} \leftarrow \{\mathcal{N}_B(\Lambda)\}$ 
3   while  $\exists p \in \mathcal{P}^{(r)} : \text{card}(p) \geq N_M$  do
4     for  $p \in \mathcal{P}^{(r)}$  do
5       if  $\text{card}(p) \geq N_M$  then Split  $p$  using
        2-means clustering with  $M$  times of
        initialization ;
6       if  $\text{card}(p) \leq N_m$  then Remove  $p$  from
         $\mathcal{P}^{(r)}$  ;
7     end
8   end
9   Compute  $\hat{U}^{(r)}(\Lambda) =$ 
     $-\sum_{p \in \mathcal{P}^{(r)}} \sum_{j=0}^1 P(C_j|p) \log(P(C_j|p))$ 
10 end
11 return  $\hat{U}(\Lambda) = \frac{1}{R} \sum_{r=1}^R \hat{U}^{(r)}(\Lambda)$ 
```

4 Proposed Method for Multi-Class Data

4.1 Outline

To enforce the Bayes error status, it is necessary and sufficient to impose an estimated boundary that satisfies the same configuration as that of the Bayes boundary. To do this, at each on-boundary sample on the estimated boundary, in accordance with Section 3, we consider only the class indexes of the two highest discriminant functions and then impose them as the class indexes of the two highest posterior probabilities. More precisely, for each estimated on-boundary sample, we want the equality between the two highest discriminant functions to correspond to the equality of the two corresponding posterior probabilities.

Therefore, even in the multi-class case, we formally come back to the two-class case because locally we consider only the two highest discriminant functions (Step 1) and then the balance between the corresponding posterior probabilities (Step 2). The procedures in the multi-class case are thus either similar to or a simple combination of the procedures given in Section 3 (Algorithm 4). Although the resulting procedure superficially looks like a concatenation of the two-class procedures from the previous sections, it inherently has a multi-class formalization.

Algorithm 4 Proposed uncertainty-measure-based method for multi-class data.

Input: Set of candidate statuses & $\mathcal{T}$

Output: $\arg \max_{\Lambda} H(\Lambda)$

```

1 for  $\Lambda \in \text{set of candidate statuses}$  do
2   Apply Algorithm 6 to obtain  $\mathcal{N}_B(\Lambda)$  (Step 1:
   Section 4.2)
3   for  $i, j \in [1, J]$  do
4     If  $\mathcal{N}_{ij}^B(\Lambda) \neq \emptyset$  then evaluate  $U_{ij}(\Lambda)$  using Eq.
       4
5     Else  $U_{ij}(\Lambda) \leftarrow 0$ 
6   end
7 end
8 Evaluate  $U(\Lambda)$  using Eq. 5 (Step2: Section 4.3)
```

4.2 Step 1: Selection of Near-Boundary Samples $\mathcal{N}_B(\Lambda)$

For the case of two classes C_0 and C_1 , Algorithm 2 used the magnitude of $g_0(\cdot; \Lambda) - g_1(\cdot; \Lambda)$ along random segments $[\mathbf{x}, \mathbf{x}']$ to generate anchors on the estimated boundary between $\hat{C}_0$ and $\hat{C}_1$. However, in the multi-class

case, if a third estimated class $\hat{C}_2$ lies between $\hat{C}_0$ and $\hat{C}_1$, then considering the difference between $g_0(\cdot; \Lambda)$ and $g_1(\cdot; \Lambda)$ no longer makes any sense (case (B) in Fig. 2). In such a case, before applying Algorithm 2, we must come back to a segment $[\mathbf{x}_a, \mathbf{x}_b]$, along which we consider only two adjacent estimated classes, and then instead consider either $g_0(\cdot; \Lambda) - g_2(\cdot; \Lambda)$ to generate anchors on $B_{02}(\Lambda)$ or $g_1(\cdot; \Lambda) - g_2(\cdot; \Lambda)$ to generate anchors on $B_{12}(\Lambda)$.

A segment $[\mathbf{x}_a, \mathbf{x}_b]$ is included in a region of adjacency between only two estimated classes, if $\forall \mathbf{x} \in [\mathbf{x}_a, \mathbf{x}_b]$ and the two highest discriminant function scores are $g_{C(\mathbf{x}_a)}(\mathbf{x}, \Lambda)$ and $g_{C(\mathbf{x}_b)}(\mathbf{x}, \Lambda)$ (this situation corresponds to case (A) in Fig. 2). In practice, we approximate this property using the two highest discriminant function scores at only $(\mathbf{x}_a + \mathbf{x}_b)/2$ (Algorithm 5). For convenience, we call this approximated property ($\mathcal{P}$). So long as ($\mathcal{P}$) is not satisfied, we halve the considered segment (case (B) in Fig. 2). Incidentally, after an anchor is produced on $B_{ij}(\Lambda)$, it is reasonable to search for only its one nearest neighbor among the near-boundary samples $\mathbf{x} \in \hat{C}_i$ or $\mathbf{x} \in \hat{C}_j$.

Algorithm 5 Near-boundary sample selection from one random sample pair for multi-class data.

Input: Random sample pair $\{a, b\} : C(a) \neq C(b)$

```

1  $\mathbf{x}_a \leftarrow a, \mathbf{x}_b \leftarrow b$ 
2 while  $[\mathbf{x}_a, \mathbf{x}_b]$  does not satisfy ( $\mathcal{P}$ ) do
3    $\mathbf{x}_b \leftarrow (\mathbf{x}_a + \mathbf{x}_b)/2$ 
4 end
5 Generate an anchor from  $[\mathbf{x}_a, \mathbf{x}_b]$  and then select the
  nearest near-boundary sample  $s$  as in Algorithm 2
6 return near-boundary sample  $s$ 
```

Algorithm 5 permits the creation of anchors on specific estimated boundaries; however, its randomness leaves little control over which boundary $B_{ij}(\Lambda)$ should create an anchor. To address this potential issue, we preliminarily sorted all samples in a matrix A , whose element A_{ij} contains the list of training samples for which the highest discriminant function score is $g_i(\cdot; \Lambda)$ and the second highest score is $g_j(\cdot; \Lambda)$ (Algorithm 6). When selecting candidates for near-boundary sample set $\mathcal{N}_{ij}^B(\Lambda)$ for C_i and C_j , we used A to preferentially form anchors based on pairs of samples $\mathbf{x}_a, \mathbf{x}_b \in A_{ij} \times A_{ji}$.

The number of anchors to be generated can be fixed similarly to that in the two-class case. The final near-boundary set $\mathcal{N}_B(\Lambda)$ is the concatenation of various sets $\{\mathcal{N}_{ij}^B(\Lambda)\}$.

Algorithm 6 Selection of $\mathcal{N}_B(\Lambda)$ for multi-class data.

Input: Classifier parameters Λ trained on $\mathcal{T}$

```

1  $\forall i, j \in [1, J], \mathcal{N}_{ij}^B(\Lambda) \leftarrow \emptyset$ 
2 Construct  $A$ 
3 for  $i \in [1, J]$  do
4   for  $j \in [1, J]$  do
5     while stop criterion not met do
6       if  $A_{ij} = \emptyset$  then break;
7       else
8         if  $A_{ji} \neq \emptyset$  then
9           Select random pair
10           $\{a, b\} \in A_{ij} \times A_{ji}$ 
11        end
12        else
13          Select random pairs
14           $\{a, b\} \in A_{ij} \times A_{ki}, k \neq j$ 
15        end
16        Denote  $s$ : output of Algorithm 5
17        provided with input  $(a, b)$ 
18        If  $s \notin \mathcal{N}_{ij}^B(\Lambda)$  then add to  $\mathcal{N}_{ij}^B(\Lambda)$ 
19      end
20    end
21  end
22 return  $\mathcal{N}_B(\Lambda) = \{\mathcal{N}_{ij}^B(\Lambda)\}_{i,j \in [1,J]^2}$ 
```

4.3 Step 2: Computation of Uncertainty Measure $U(\Lambda)$

In the multi-class case, it is possible that the class indexes between the two highest local discriminant function scores are different from the indexes between the two highest local posterior probabilities. Therefore, we need to define an uncertainty measure that handles this situation.

Assuming that the two highest local discriminant function scores correspond to $\hat{C}_i$ and $\hat{C}_j$, we define uncertainty measure $H_{ij}(\mathbf{x})$ as follows. Given sample $\mathbf{x}$ on $B_{ij}(\Lambda)$, we first choose the two highest posterior probabilities $P(C_{(i)}|\mathbf{x})$ and $P(C_{(ii)}|\mathbf{x})$, and next we set $H_{ij}(\mathbf{x})$ in the following conditional branching manner:

- I. If $\{(i), (ii)\} \neq \{i, j\}$, then set $H_{ij}(\mathbf{x})$ to 0.
- II. Else normalize $P(C_{(i)}|\mathbf{x})$ and $P(C_{(ii)}|\mathbf{x})$ so that their sum equals 1, and then apply (2) to the normalized posterior probabilities by replacing class indexes $\{0, 1\}$ with $\{(i), (ii)\}$. Here, normalization is necessary because the definition of Eq. 2 assumes $P(C_{(i)}|\mathbf{x}) + P(C_{(ii)}|\mathbf{x}) = 1$, which does not necessarily hold.

The resulting normalized entropy is similar to the concept of entropy branching. In our above definition, the first step means that if the sample is not even on a region near (true) classes C_i and C_j , then by default we set the entropy to its lowest value 0.

Then, we define a local uncertainty measure score for $B_{ij}(\Lambda)$ similarly to Eq. 3:

$$U_{ij}(\Lambda) = \frac{1}{N_{ij}^B} \sum_{\mathbf{x} \in \mathcal{N}_{ij}^B(\Lambda)} H_{ij}(\mathbf{x}), \quad (4)$$

where N_{ij}^B is the number of samples in $\mathcal{N}_{ij}^B(\Lambda)$. Finally, a multi-class uncertainty measure score $U(\Lambda)$ is considered simply the mean over several local uncertainty scores $U_{ij}(\Lambda)$:

$$U(\Lambda) = \frac{1}{S} \sum_{i,j \in J} N_{ij}^B U_{ij}(\Lambda), \quad (5)$$

where $S = \sum_{i,j} N_{ij}^B$. We accordingly select the classifier status Λ that provides the highest $U(\Lambda)$. Under these conditions, the partitioning of each near-boundary set $\mathcal{N}_{ij}^B(\Lambda)$ and the estimation of posterior probabilities are performed exactly as described in Section 3.3.

5 Optimality Analysis of Uncertainty Measure

5.1 Overview and Preparations

As can be seen from the definition in Sections 3 and 4, the reliability of our proposal relies on the quality of the computed uncertainty measure. Now, we mathematically analyze our entropy-based procedure for computing uncertainty measure scores and show how our uncertainty-measure-based method can reliably represent Bayes boundary-ness to find the optimal classifier parameter status, even if it uses only training samples in a single-shot manner.

We consider point $\mathbf{x}$ in a finite-dimensional Euclidian space; furthermore let $X_1, \dots, X_N$ be N samples independently sampled from a probability distribution function (pdf) $p(\cdot)$.² Then, every point in the space is assumed to belong to one or multiple classes among J classes $C_1, \dots, C_J$.³ Moreover, letting $\mathcal{R}^{(N)}(\mathbf{x})$ and $V^{(N)}(\mathbf{x})$ be a small region containing $\mathbf{x}$ and the volume of this region, respectively, we assume the following:

- I. The distance between any two points in $\mathcal{R}^{(N)}(\mathbf{x})$ goes to 0 as $N \rightarrow \infty$.
- II. pdf $p(\cdot)$ and class likelihoods $p(\cdot|C_j)$ ($j = 1, \dots, J$) are continuous functions.

²The term “sample” means random variable in this subsection.

³Class label (index) is basically a random variable in this subsection.

III. For all N and all $\mathbf{x}$, $\mathcal{R}^{(N)}(\mathbf{x})$ is a finite and closed set.

Note that we make no assumption about the shape of $\mathcal{R}^{(N)}(\mathbf{x})$.

5.2 Convergence to True Value for Ratio-Based Probability Density Estimator

First, the probability $P^{(N)}(\mathbf{x})$ that sample X is contained in $\mathcal{R}^{(N)}(\mathbf{x})$ is equal to the expected value of $1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)$:

$$P^{(N)}(\mathbf{x}) = \mathbb{E}[1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)] = \int_{\mathcal{R}^{(N)}(\mathbf{x})} p(u) du, \quad (6)$$

where $1_A(\cdot)$ refers to the indicator function that outputs 1 if the predicate A holds true, and 0 otherwise. Then, there exists point $\mathbf{x}^{(N)} \in \mathcal{R}^{(N)}(\mathbf{x})$ such that $P^{(N)}(\mathbf{x}) = p(\mathbf{x}^{(N)}) \cdot V^{(N)}(\mathbf{x})$ (see Appendix 1). Furthermore, the variance of $1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)$ is

$$\begin{aligned} \text{Var}[1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)] &= \mathbb{E}[\{1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)\}^2] - \{P^{(N)}(\mathbf{x})\}^2 \\ &= \mathbb{E}[1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)] - \{P^{(N)}(\mathbf{x})\}^2 \\ &= P^{(N)}(\mathbf{x})\{1 - P^{(N)}(\mathbf{x})\}. \end{aligned} \quad (7)$$

For the N i.i.d. samples, we next denote the number of samples contained in $\mathcal{R}^{(N)}(\mathbf{x})$ by $K^{(N)}(\mathbf{x})$. Then, $K^{(N)}(\mathbf{x})$ is represented as

$$K^{(N)}(\mathbf{x}) = \sum_{n=1}^N 1_{\mathcal{R}^{(N)}(\mathbf{x})}(X_n). \quad (8)$$

Because $X_1, \dots, X_N$ are i.i.d. samples, the expectation and variance of $K^{(N)}(\mathbf{x})$ are

$$\begin{aligned} \mathbb{E}[K^{(N)}(\mathbf{x})] &= N \mathbb{E}[1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)] \\ &= N P^{(N)}(\mathbf{x}), \end{aligned} \quad (9)$$

$$\begin{aligned} \text{Var}[K^{(N)}(\mathbf{x})] &= N \text{Var}[1_{\mathcal{R}^{(N)}(\mathbf{x})}(X)] \\ &= N P^{(N)}(\mathbf{x})\{1 - P^{(N)}(\mathbf{x})\}, \end{aligned} \quad (10)$$

and accordingly the following holds:

$$\begin{aligned} \mathbb{E}\left[\frac{K^{(N)}(\mathbf{x})}{N V^{(N)}(\mathbf{x})}\right] &= \frac{1}{N V^{(N)}(\mathbf{x})} \mathbb{E}[K^{(N)}(\mathbf{x})] \\ &= p(\mathbf{x}^{(N)}), \end{aligned} \quad (11)$$

$$\begin{aligned} \text{Var}\left[\frac{K^{(N)}(\mathbf{x})}{N V^{(N)}(\mathbf{x})}\right] &= \frac{N P^{(N)}(\mathbf{x})\{1 - P^{(N)}(\mathbf{x})\}}{\{N V^{(N)}(\mathbf{x})\}^2} \\ &= \frac{p(\mathbf{x}^{(N)})\{1 - P^{(N)}(\mathbf{x})\}}{N V^{(N)}(\mathbf{x})}. \end{aligned} \quad (12)$$

If $V^{(N)}(\mathbf{x})$ is chosen so that it satisfies $V^{(N)}(\mathbf{x}) \rightarrow 0$ and $N V^{(N)}(\mathbf{x}) \rightarrow \infty$,⁴ then from the continuity of $p(\cdot)$ it

⁴For example, given $V > 0$, we can set $V^{(N)}(\mathbf{x})$ as $V/(N^\alpha)$ ($0 < \alpha < 1$) or $V/(\ln N)^\beta$ ($N > 1, \beta > 0$), etc.

follows that

$$p(\mathbf{x}^{(N)}) \rightarrow p(\mathbf{x}) \quad (N \rightarrow \infty); \quad (13)$$

furthermore, from Eqs. 11 and 12, we reach

$$\mathbb{E} \left[\frac{K^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] \rightarrow p(\mathbf{x}), \text{Var} \left[\frac{K^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] \rightarrow 0 \quad (N \rightarrow \infty). \quad (14)$$

Therefore, $K^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}$ converges to $p(\mathbf{x})$ in the mean-square (L_2) sense, and in particular to $p(\mathbf{x})$ in probability:

$$\frac{K^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \xrightarrow{P} p(\mathbf{x}) \quad (N \rightarrow \infty), \quad (15)$$

where $\xrightarrow{P}$ denotes convergence in probability.

5.3 Convergence to True Value for Ratio-Based Joint Probability Density Estimator

The reasoning in this subsection is similar to that in the previous subsection.

The data available for training is a set of N realizations $\{\mathbf{x}_1, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N\}$, where $\mathbf{x}_n$ is a realization of $\mathbf{X}_n$ that is deterministically associated with class label (index) y_n . On the other hand, each random variable $\mathbf{X}_n$ ($n = 1, \dots, N$) is probabilistically associated with multiple class labels, since different class regions can overlap in the Euclidean sample space. Therefore, we regard a class label for $\mathbf{X}_n$ ($n = 1, \dots, N$) as a random variable Y_n ($\in \{1, \dots, J\}$) and discuss sample pairs $\{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n), \dots, (\mathbf{X}_N, Y_N)\}$ as follows, assuming that these sample pairs are independent.

For one sample pair $(\mathbf{X}, Y)$, we denote the expectation of $1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)$ by $P_j^{(N)}(\mathbf{x})$. Then, this expectation is the probability that $\mathbf{X}$ is included in $V^{(N)}(\mathbf{x})$ and labeled by C_j (see Appendix 2):

$$P_j^{(N)}(\mathbf{x}) = \mathbb{E} [1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)] \\ = \Pr(C_j) \int_{\mathcal{R}^{(N)}(\mathbf{x})} p(\mathbf{u}|C_j) d\mathbf{u}. \quad (16)$$

Moreover, because $p(\cdot|C_j)$ is continuous, there exists point $\mathbf{x}^{(N,j)} \in \mathcal{R}^{(N)}(\mathbf{x})$, and $P_j^{(N)}(\mathbf{x})$ is equal to

$\Pr(C_j)p(\mathbf{x}^{(N,j)}|C_j) \cdot V^{(N)}(\mathbf{x})$ (see Appendix 1); the variance of $1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)$ can be written as

$$\begin{aligned} & \text{Var} [1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)] \\ &= \mathbb{E} [\{1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)\}^2] - \{P_j^{(N)}(\mathbf{x})\}^2 \\ &= \mathbb{E} [1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}) \cdot 1_{\{j\}}(Y)] - \{P_j^{(N)}(\mathbf{x})\}^2 \\ &= P_j^{(N)}(\mathbf{x})\{1 - P_j^{(N)}(\mathbf{x})\}. \end{aligned} \quad (17)$$

For the N i.i.d. samples, we denote the number of samples included in $\mathcal{R}^{(N)}(\mathbf{x})$ and labeled by C_j as $K_j^{(N)}(\mathbf{x})$. Then, $K_j^{(N)}(\mathbf{x})$ is represented as

$$K_j^{(N)}(\mathbf{x}) = \sum_{n=1}^N 1_{\mathcal{R}^{(N)}(\mathbf{x})}(\mathbf{X}_n) \cdot 1_{\{j\}}(Y_n). \quad (18)$$

Moreover, because $(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_N, Y_N)$ are i.i.d., the expectation and variance of $K_j^{(N)}(\mathbf{x})$ are

$$\mathbb{E} [K_j^{(N)}(\mathbf{x})] = NP_j^{(N)}(\mathbf{x}), \quad (19)$$

$$\text{Var} [K_j^{(N)}(\mathbf{x})] = NP_j^{(N)}(\mathbf{x})\{1 - P_j^{(N)}(\mathbf{x})\}, \quad (20)$$

and it follows that

$$\mathbb{E} \left[\frac{K_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] = \frac{NP_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} = \Pr(C_j)p(\mathbf{x}^{(N,j)}|C_j), \quad (21)$$

$$\begin{aligned} \text{Var} \left[\frac{K_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] &= \frac{NP_j^{(N)}(\mathbf{x})\{1 - P_j^{(N)}(\mathbf{x})\}}{\{NV^{(N)}(\mathbf{x})\}^2} \\ &= \frac{\Pr(C_j)p(\mathbf{x}^{(N,j)}|C_j)\{1 - P_j^{(N)}(\mathbf{x})\}}{NV^{(N)}(\mathbf{x})}. \end{aligned} \quad (22)$$

Here, if we set $V^{(N)}(\mathbf{x})$ similarly to that used to derive (13), and then we reach the following, since $p(\cdot|C_j)$ is a continuous function:

$$\mathbb{E} \left[\frac{K_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] \rightarrow \Pr(C_j)p(\mathbf{x}|C_j), \text{Var} \left[\frac{K_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \right] \rightarrow 0 \quad (N \rightarrow \infty). \quad (23)$$

Accordingly, $K_j^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}$ converges to $\Pr(C_j)p(\mathbf{x}|C_j)$ in the mean-square sense, and in particular to $\Pr(C_j)p(\mathbf{x}|C_j)$ in probability:

$$\frac{K_j^{(N)}(\mathbf{x})}{NV^{(N)}(\mathbf{x})} \xrightarrow{P} \Pr(C_j)p(\mathbf{x}|C_j) \quad (N \rightarrow \infty). \quad (24)$$

5.4 Probabilistic Convergence to True Value for Ratio-Based Posterior Probability Estimator

In previous subsections, we found that the simple ratio-based estimators for the probability density and the joint probability density converge to their true values in probability, respectively (see Eqs. 15 and 24). From these

results and the nature of four arithmetic operations for the random variable sequences that converge in probability, we finally obtain

$$\begin{aligned} \frac{K_j^{(N)}(\mathbf{x})}{K^{(N)}(\mathbf{x})} &= \frac{K_j^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}}{K^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}} \\ &\xrightarrow{P} \frac{\Pr(C_j)p(\mathbf{x}|C_j)}{p(\mathbf{x})} = \Pr(C_j|\mathbf{x}) \quad (N \rightarrow \infty) \end{aligned} \quad (25)$$

Because the above convergences in Eqs. 13 through 25 assume that $V^{(N)}(\mathbf{x}) \rightarrow 0$ ($N \rightarrow \infty$), the following should be satisfied for $K^{(N)}(\mathbf{x})$, $K_j^{(N)}(\mathbf{x})$ and N :

- I. If $p(\mathbf{x}), p(\mathbf{x}|C_j) > 0$, then $K^{(N)}(\mathbf{x}), K_j^{(N)}(\mathbf{x}) \rightarrow \infty$ ($N \rightarrow \infty$).
- II. $K^{(N)}(\mathbf{x})/N, K_j^{(N)}(\mathbf{x})/N \rightarrow 0$ ($N \rightarrow \infty$).

The first condition is necessary because if $K^{(N)}(\mathbf{x})$ and $K_j^{(N)}(\mathbf{x})$ do not go to infinity when $N \rightarrow \infty$, then they would tend to 0 for sufficiently small values of $V^{(N)}(\mathbf{x})$. The second condition ensures that $K^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}$ and $K_j^{(N)}(\mathbf{x})/\{NV^{(N)}(\mathbf{x})\}$ do not diverge even when the first condition is satisfied.

5.5 Practical Advantages Supported by Optimality in Ratio-Based Posterior Probability Estimation

In addition to the property of convergence to the true posterior probability in Eq. 25, the formalization eloquently expresses the practical advantages, which will be useful in a real-life finite sample regime, of our own k NN-based posterior probability estimation using only near-boundary samples. Assuming that point $\mathbf{x}$ is sufficiently close to estimated boundary $B(\Lambda)$, we summarize them in the following:

- I. We should decrease region $\mathcal{R}^{(N)}(\mathbf{x})$ to reduce the bias in the expectation of $K_j^{(N)}(\mathbf{x})/K^{(N)}(\mathbf{x})$ so that $p(\mathbf{x}^{(N)})$ and $p(\mathbf{x}^{(N,j)}|C_j)$ become closer to $p(\mathbf{x})$ in Eq. 11 and $p(\mathbf{x}|C_j)$ in Eq. 21, respectively.
- II. By increasing $\mathcal{R}^{(N)}(\mathbf{x})$, we can basically reduce the variance of k NN-based posterior probability estimate $K_j^{(N)}(\mathbf{x})/K^{(N)}(\mathbf{x})$. This result can also be proved in a more accurate manner using the perturbation technique [2].
- III. When estimated boundary $B(\Lambda)$ is close to the Bayes boundary, we can reduce both the bias and the variance, based on Eqs. 11, 12, 21, and 22, in the posterior probability estimate by applying region $\mathcal{R}^{(N)}(\mathbf{x})$ to only the near-boundary samples in $\mathcal{N}_B(\Lambda)$ and increasing its size (in other words, increasing the number of the near-boundary samples used for k NN-based posterior probability estimation). Here, all individual estimates $P(C_j|\mathbf{x})$ will be close to 0.5,

and, moreover, the numerators in Eqs. 12 and 22 are bounded by $N/4$; therefore, the corresponding variances are bounded by $1/\{4N(V^{(N)}(\mathbf{x}))^2\}$ and tend to 0 as $V^{(N)}(\mathbf{x})$ grows larger. This valuable property enables our method to use large regions (clusters) for the posterior probability estimation in Algorithm 3 and leads to accurate and reliable performances even if it adopts a simple k NN-based estimation.

- IV. When accepting the incursion of samples outside $\mathcal{N}_B(\Lambda)$ to region $\mathcal{R}^{(N)}(\mathbf{x})$ and increasing its size, we clearly face a dilemma between the bias and the variance: Increasing $\mathcal{R}^{(N)}(\mathbf{x})$ decreases the variance but increases the bias, while decreasing $\mathcal{R}^{(N)}(\mathbf{x})$ increases the variance but decreases the bias. This phenomenon, which is generally observed in a regular k NN-based posterior probability estimation, proves again the validity of using only near-boundary samples for posterior probability estimation.
- V. All the estimators' properties such as the convergence to true value hold regardless of the shape of region $\mathcal{R}^{(N)}(\mathbf{x})$.

Usually, k NN-based posterior probability estimation controls the size and shape of a region whose samples are used for estimation, and basically it cannot avoid the tradeoff between bias and variance in the estimation. In contrast, our k NN-based method can basically reduce both bias and variance using only the near-boundary samples in a small region, which can be extended along an estimated boundary, for the estimation.

6 Experimental Evaluation

6.1 Datasets

We conducted evaluations on fixed-dimensional vector pattern datasets from the UCI Machine Learning Repository.⁵ Especially for the Abalone, Wine Quality Red, and Wine Quality White datasets, we used our custom versions, where the original categories were grouped into three categories due to the presence of very few represented classes. For the Wine Quality White dataset, we used a randomly sampled subset of the mother dataset. For analysis purposes, we also prepared a two-dimensional two-class synthetic vector pattern dataset called GMM, which modeled each class with two Gaussian mixtures and 1100 samples. We summarize these datasets in Table 1, where N refers to the number of samples available, D to the dimensionality, and J to the number of classes.

⁵<http://archive.ics.uci.edu/ml/index.php>

Table 1 Datasets.

Dataset	N	D	J	Remarks
GMM	2,200	2	2	synthetic data
Abalone	4,177	7	3	custom version
Breast Cancer	683	9	2	2:1 imbalance
Cardiotocography	1,831	30	2	10:1 imbalance
Ionosphere	351	34	2	2:1 imbalance
Letter Recognition	20,000	16	26	
MNIST (test)	10,000	784	10	
Landsat Satellite	6,435	36	7	
Sonar (all data)	208	60	2	
Spambase	4,601	57	2	
Thyroid	7,200	21	3	18:1:1 imbalance
Wine Quality Red	1599	11	3	custom version
Wine Quality White	1470	11	3	custom version

For all datasets, we performed sample vector normalization, i.e. removing the mean and scaling to unit variance, in a vector-element-by-vector-element manner.

6.2 Classifier

As the classifier for evaluation, we chose SVM [27] using a Gaussian kernel whose implementation is available online.⁶ In this case, Λ consists of a set of kernel weights optimized during the training, regularization parameter C , and Gaussian kernel width γ . To simplify the analysis, we fixed C beforehand using another CV-based preliminary experiment and then focused on the optimal setting (status selection) of the single parameter γ .

For multi-class data, we used a one-versus-all multi-class SVM. Actually, the one-versus-all formalization of the multi-class problem is different from the multi-class formalization described in Section 2. However, in our understanding, the SVM implementation that we used draws boundaries such that a region near the estimated boundary $B_{ij}(\Lambda)$ is characterized by $g_i(\cdot; \Lambda)$ and $g_j(\cdot; \Lambda)$, being the two highest discriminant function scores. This is adequate for Step 1, described in Section 4.2, to be applicable.

6.3 Evaluation Procedure

To evaluate the effectiveness of our uncertainty-measure-based boundary evaluation method, we need to use the Bayes boundary as a reference or true target for the adopted datasets. However, the Bayes boundary is rarely known for real-life data. Our comparison competitor, the CV-based method, compensated for this lack of information. To obtain accurate estimates of the Bayes boundary, we

applied an SVM classifier to each adopted dataset in the CV manner, computed two kinds of classification error probability estimates, i.e. averaged estimate for the CV's training sample folds (L_{tr}) and averaged estimate for the CV's validation (testing) sample folds (L_{val}), for every setting of hyperparameter γ , and used the values of L_{val} as nearly true targets to be compared to our uncertainty measure values; therefore, we treated the minimum of L_{val} as a nearly true Bayes error value and its corresponding boundary estimate as a nearly true Bayes boundary.

For the above purpose, we applied five-fold CV to each of the adopted datasets, except for Breast Cancer, Ionosphere, and Sonar, for which we applied LOO-CV owing to the few samples available. In both the CV and LOO-CV procedures, we divided all of the samples for each dataset into training folds and validation folds. Moreover, to check the reliability of the targets, we preliminarily tested several different settings of fold numbers and selected the five-fold setting that produced sufficient reliability.

To compute the uncertainty measure, we used all of the samples for each dataset ($\mathcal{T}$ = all of the given samples), differently from the CV-based method. The computed uncertainty measure $U(\Lambda)$ is high when the estimated boundary is close to the Bayes boundary; conversely, the target value L_{val} becomes lower. For convenience of discussion, we use a sign-reversed measure $-U(\Lambda)$ and analyze the similarity between $-U(\Lambda)$ and L_{val} in later sections.

It is known that the quality of posterior probability estimation can quickly degrade when the samples are imbalanced in classes, probably due to the erroneous sampling from the mother set consisting of an infinite number of samples or to the imbalance in prior probabilities [26]. Traditional solutions to the presence of an under-represented minority class modify the classifier objective or resample the minority-class samples to force class balance

⁶<http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

[26]. In our method, rather than changing the samples or the classification process, we simply evaluate classification uncertainty in a non-intrusive way. Our solution is to superficially take the class imbalance into account during the posterior probability estimation. More precisely, we replace the estimated posterior probabilities $\hat{P}(C_i|\mathbf{x})$ with $\hat{P}(C_i|\mathbf{x})/\hat{P}(C_i)$ in Eq. 2. By doing so, we will raise the estimated posterior probability for a low prior probability, while we will bring down the estimated posterior probability for a high prior probability. The next sections assume this simple measure.

6.4 Hyperparameters

Our proposed method contains several hyperparameters for controlling its process. Despite the presence of several such hyperparameters in our method, these hyperparameters do not need dataset-by-dataset tuning, and they can be quite insensitively set.

In Algorithm 2, we simply set the maximum number of dichotomy loops to 30 and the maximum number of repetitions l_M for generating anchors to a high value such as 10,000.

In Algorithm 3, N_m and N_M control the granularity of the clusters, each of which should ideally have as small a volume as possible and yet contain enough samples to perform a reliable entropy (or posterior probability) estimation. Computing the entropy based on clusters containing two or three samples is obviously rough, so we imposed clusters that would contain 10 or fewer samples by simply setting $N_m = 8$ and $N_M = 12$. Moreover, initialization count M for each hierarchical clustering and count R of repeating the clustering should simply be higher than 1 to increase the reliability of the procedure; accordingly, we simply set them to 10.

6.5 Results and Discussions

6.5.1 Selection of $\mathcal{N}_B(\Lambda)$

We first show the results obtained by Algorithm 2 with five different settings of γ on the GMM dataset (Fig. 3). A higher γ should correspond to a more complex boundary (a lower one for a simpler boundary). The results clearly show that the selected near-boundary samples were accurately selected along the estimated boundary. For excessively low

Figure 3 Results obtained by our near-boundary sample selection step (Algorithm 2) with five different settings of γ on two-class GMM dataset. From left to right and up to down: near-boundary samples (blue crosses) and corresponding training samples with estimated class

labels (gray or green) for $\gamma = 2^{-35}, 2^{-22}, 2^{-9}, 2^{-1}, 2^{14}$. Training samples with original class labels (gray and red) shown in bottom-right corner.

values like $\gamma = 2^{-35}$, 2^{-22} , the estimated boundaries were too simple and vague; the selected near-boundary samples are allocated along straight lines or bands. For excessively high values like $\gamma = 2^{-1}$, 2^{14} , the estimated boundaries were very complex; the selected near-boundary samples are scattered. For $\gamma = 2^{-9}$, which was not necessarily the best but for the most part appropriately selected, the estimated boundary and its corresponding near-boundary samples almost exactly trace the true Bayes boundary.

6.5.2 Overview of Results on Classifier Status Selection

In Fig. 4, for each adopted dataset, we show L_{tr} (yellow curve), L_{val} (green curve), and our proposed uncertainty measure score $-U(\Lambda)$ (blue curve). Note that we obtained the blue curve of $-U(\Lambda)$ using all of the samples of each

dataset for training. In all of the panels of these figures, the horizontal axis corresponds to the value of hyperparameter γ (kernel width) as well as its corresponding classifier parameter status (the values of SVM's kernel weights); for every different value of γ , we trained the SVM classifier and obtained a set of trained kernel weights. Because we assume that L_{val} is an accurate estimate of the error probability, its minimum represents the Bayes boundary and its corresponding optimal status of kernel width γ and kernel weights. Therefore, our desired result for the blue curve of $-U(\Lambda)$ is to reach a minimum at the same value of γ as the green curve of L_{val} .

For all datasets, the yellow curve of L_{tr} goes down as γ increases; for lower boundary complexity (low γ), the yellow curve of L_{tr} is close to the green curve of L_{val} , but the gap between the yellow and green curves grows wider

Figure 4 Classifier parameter status selection results for synthetic GMM dataset and 11 real-life datasets. From top to bottom: GMM, Abalone, Breast Cancer, Cardiotocography, Ionosphere, and Landsat Satellite. Each panel shows the estimated error probability curve based on the CV's training sample folds (L_{tr} , yellow), the estimated error probability curve based on the CV's validation sample folds (L_{val} , green), and our uncertainty measure curve ($-U$, blue). In the top GMM panel, the red curve (L_{val2}) represents the estimated posterior probability over the 20,000 independent testing samples. Horizontal axis indicates the value of γ . From top to bottom: Letter Recognition, MNIST (test), Sonar, Spambase, Wine Quality Red, and Wine Quality White.

and wider as the classifier draws more complex boundaries (higher γ). Such a gap clearly illustrates the phenomenon of overlearning, or the underestimation of the Bayes error.

In particular, for the GMM dataset, we additionally generated 20,000 independent samples using the same Gaussian mixture model as that used for the original 1100 samples, approximated the true error probabilities, which should originally be computed over an infinite number of samples, and showed its results as the red curve (L_{val2} of Fig. 4). Interestingly, the green CV-based curve and the red large-data-based curve closely fit each other. This consistent result shows that the CV-based method accurately approximated the true error probabilities and reliably served as a competitor or target in our comparison.

For all datasets, the blue curve of our uncertainty measure $-U$ almost always shows the same trend as the green curve of the CV-based error probability estimates; in particular, the minimum of the blue curve is among the bottom points of the green curve, which should correspond to the optimal classifier parameter status. As found in the values of L_{val} , classification for the Abalone, Landsat Satellite, Wine Quality Red, and Wine Quality White datasets was rather difficult; for these datasets, the Bayes error values estimated by the CV-based method were higher than 0.3. Even for these difficult datasets, our blue curve closely followed the green curve. From all of these comparison results, we can infer the basic utility of our uncertainty-measure-based method for selecting the optimal classifier parameter status.

6.5.3 Influence of Data Imbalance on Classifier Status Selection

To better understand the influence of class imbalance, we focus on the Cardiotocography dataset (176 samples for C_0 and 1,655 samples for C_1) in Fig. 5. We break down the number of near-boundary samples (black curve) into the numbers of near-boundary samples belonging to C_0 (gray curve) and to C_1 (red curve). As γ increases, the number of near-boundary samples belonging to C_0 rapidly increases, stays high, and drastically decreases in $\mathcal{N}_B(\Lambda)$, while the number of samples belonging to C_1 almost always stays around 176; for most γ values, the total number of near-boundary samples is dominated by one of the two classes. In this case, the computation of the uncertainty measure is obviously biased and there is no way that uncertainty around B^* can be achieved. The posterior probability computation with the superficial prior probability correction described in Section 6.3 indeed gives better results (blue curve) than without the prior correction (red dashed curve); however, more efficient posterior estimation methods must be applied for such an imbalanced case.

Figure 5 Parameter status selection results for Cardiotocography data. In the top panel, the blue curve represents uncertainty measure $-U$ with the prior probability correction; the red dashed curve for $-U$ without the correction. In the bottom panel, the black curve represents the number of near-boundary samples; the gray and red curves represent the numbers of near-boundary samples belonging to C_0 and to C_1 , respectively.

6.5.4 Influence of Neighbor Selection

To further analyze the parameter status selection results in Section 6.5.2, we performed a step-by-step analysis of Step 2, which executes the posterior probability estimation using the k NN. However, instead of the traditional estimation based on a fixed-size neighborhood selected from the entire $\mathcal{T}$, we chose to consider only the near-boundary samples in $\mathcal{N}_B(\Lambda)$ as neighbors and also determined the number of neighbors adaptively by the hierarchical clustering procedure.

Figure 6 Comparison of three neighbor selection schemes in Step 2 (near-boundary sample selection) for Ionosphere dataset: fixed k NN considering neighbors selected from $\mathcal{T}$ (top panel), fixed k NN considering neighbors selected from $\mathcal{N}_B(\Lambda)$ (middle panel), and adaptive partitioning in $\mathcal{N}_B(\Lambda)$ (bottom panel).

Figure 7 Effect of using different k in applying k NN to $\mathcal{T}$ for the Ionosphere dataset. L_{val} is shown by green dashed line. Curves for $k = 5, 7, 10, 15$ are shown in red, green, orange, and blue, respectively.

In this section, we analyze the influence of this choice on the quality of the estimation of posterior probabilities along the estimated boundary. To this end, we compared the overall parameter status selection results obtained on the Ionosphere dataset by three neighbor-selection schemes (Fig. 6): fixed k NN considering neighbors selected from the entire $\mathcal{T}$ (top panel), fixed k NN considering neighbors selected from $\mathcal{N}_B(\Lambda)$ (middle panel), and adaptive partitioning in $\mathcal{N}_B(\Lambda)$ (bottom panel). For the top and middle panels, we first fixed k to 5 (results shown using different values of k). The setting of N_m, N_M, r, R, p_{ij} for the adaptive partitioning was the same as in Section 6.5.2.

First, the top panel basically shows noise: there is no trend and the range of values for $-U(\Lambda)$ is close to 0 (see the right vertical axis), which means the method does not measure any reliable uncertainty measure score around $B(\Lambda)$, even when $B(\Lambda)$ is close to B^* . By contrast, the middle panel seems to clearly detect a range of suitable candidate parameter values similar to the range provided by the CV-based method. The neat improvements from the top panel to the middle panel show the necessity of filtering out samples outside of $\mathcal{N}_B(\Lambda)$. Here, note that the range of $-U(\Lambda)$ in the top panel is significantly smaller than that in the middle and bottom panels. Accordingly, these results can be understood as follows. To evaluate the classifier status, our measure strictly focuses on estimation of the posterior probability imbalance *on the estimated boundary*. In practice, with finite samples, $\mathcal{N}_B(\Lambda)$ contains only samples close to $B(\Lambda)$. Step 1 can be seen as a

Figure 8 Effect of using different k in applying k NN to $\mathcal{N}_B(\Lambda)$ for the Ionosphere dataset. L_{val} is shown by green dashed line. Curves for $k = 5, 7, 10, 15$ are shown in red, green, orange, and blue, respectively.

Figure 9 Influence of M and R on the reliability of classifier parameter selection for the Ionosphere dataset. From top to bottom: $\{M = 1, R = 1\}$, $\{M = 10, R = 1\}$, $\{M = 1, R = 10\}$, $\{M = 20, R = 20\}$.

filter that effectively cuts noise, i.e. samples away from the estimated boundary.

Second, the clear improvement from the middle panel to the bottom panel shows that the adaptive partition of the near-boundary samples further enhances the quality of the neighbors used in the k NN-based estimation. Intuitively, adapting the number of neighbors to the local density and ignoring excessively small neighbors improves the quality of the posterior estimation.

For exhaustiveness, using the Ionosphere dataset, we also tried $k = 5, 7, 10, 15$ when using the k NN posterior estimation in the top and middle panels of Fig. 6, and we summarized the results in Fig. 7 (for top panel of Fig. 6) and Fig. 8 (for middle panel of Fig. 6). Compared to the choice of k for $\mathcal{T}$, the choice of k for the near-boundary sample set $\mathcal{N}_B(\Lambda)$ clearly has a minor impact on the quality of the uncertainty measure computation.

6.5.5 Influence of Procedure Repetition on Classifier Status Selection

Figure 9 shows the effect of increasing the number of random trials for the 2-means clustering M and the number of near-boundary set partitions R over the reliability of the selection procedure. As can be seen, further repetition gives a neater blue curve, which is closer to the target green curve. The increase in similarity is striking, all the more so because the two curves are obtained from apparently independent evaluation criteria, i.e. classification error probability and uncertainty measure. Such results seem to confirm the following two conclusions. First, the results from CV and

our procedure mutually confirm their reliability as methods of classifier parameter status evaluation. Second, increasing M and R increases the reliability of the posterior balance estimate on $B(\Lambda)$. Although the increase in reliability is not negligible, even $M = 1$, $R = 1$ seems sufficient, at least for the Ionosphere dataset.

7 Conclusion

We introduced a new method to evaluate a general form of classifier. The purpose was to define a new way of selecting the optimal classifier parameter status to overcome the fundamental limitations of the standard methods (i.e. training repetition, data splitting in validation, difficulty in applying them). Accordingly, we defined a fundamentally different classifier evaluation criterion that directly takes the Bayes boundary status as a reference, and that can potentially be estimated without the need for validation data, while still being readily applicable to any type of classifier and fixed-dimension data. Moreover, we mathematically proved the validity of our posterior probability estimation procedure, which plays a central role in the proposed method for finding the optimal classifier parameter status. The experimental results and a comparison with the benchmark CV method indicate the possibility of selecting the optimal model on several real-life classification tasks. Despite the encouraging results, there is room for improvement in terms of accuracy and processing speed. To improve accuracy, our mathematical analysis of the posterior probability estimation along the estimated boundary may provide guidelines on the selection and use of the boundary neighborhood. To improve processing speed, we believe that a better use of the information provided by the discriminant functions and of the local information along the estimated boundary can simplify the current somewhat complicated process of Step 1. Taken together, these improvements would provide a practical competitive edge to our method, especially for expensive classifier training.

Acknowledgments This work was supported in part by JSPS KAKENHI No.18H03266 and MEXT Supported Program for Strategic Research Foundation at Private Universities (2014-2018), named “Driver-in-the-Loop.”

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Appendix: 1

Because $p(\cdot)$ is continuous, it has minimum value m and maximum value M within the bounded and closed set $\mathcal{R}^{(N)}(\mathbf{x})$:

$$m \leq p(\mathbf{u}) \leq M \quad (\mathbf{u} \in \mathcal{R}^{(N)}(\mathbf{x})).$$

Then, integration over $\mathcal{R}^{(N)}(\mathbf{x})$ gives

$$m \cdot V^{(N)}(\mathbf{x}) \leq \int_{\mathcal{R}^{(N)}(\mathbf{x})} p(\mathbf{u}) d\mathbf{u} \leq M \cdot V^{(N)}(\mathbf{x}),$$

and thus

$$m \leq \frac{P^{(N)}(\mathbf{x})}{V^{(N)}(\mathbf{x})} \leq M.$$

Because m and M correspond to values taken by $p(\cdot)$ on $\mathcal{R}^{(N)}(\mathbf{x})$, the intermediate value theorem guarantees the existence of $\mathbf{x}^{(N)} \in \mathcal{R}^{(N)}(\mathbf{x})$, which satisfies

$$\frac{P^{(N)}(\mathbf{x})}{V^{(N)}(\mathbf{x})} = p(\mathbf{x}^{(N)}).$$

Appendix: 2

We denote the probability space that governs the random variables by $(\Omega, \mathcal{F}, \Pr)$, where Ω is the space of all events, $\mathcal{F}$ is a completely additive class over Ω , and $\Pr$ is a probability measure over $\mathcal{F}$. Then, the following holds:

$$\begin{aligned} P_j^{(N)}(\mathbf{x}) &= \mathbb{E} [1_{\mathcal{R}^{(N)}(\mathbf{x})}(X) \cdot 1_{\{j\}}(Y)] \\ &= \int_{\Omega} 1_{\mathcal{R}^{(N)}(\mathbf{x})}(X(\omega)) \cdot 1_{\{j\}}(Y(\omega)) \Pr(d\omega) \\ &= \int_{\Omega} 1_{X^{-1}(\mathcal{R}^{(N)}(\mathbf{x})) \cap Y^{-1}(\{j\})}(\omega) \Pr(d\omega) \\ &= \Pr(X^{-1}(\mathcal{R}^{(N)}(\mathbf{x})) \cap Y^{-1}(\{j\})) \\ &= \Pr\{X \in \mathcal{R}^{(N)}(\mathbf{x}) \cap Y = j\} \\ &= \Pr\{Y = j\} \cdot \Pr(\{X \in \mathcal{R}^{(N)}(\mathbf{x})\} | \{Y = j\}) \\ &= \Pr(C_j) \int_{\mathcal{R}^{(N)}(\mathbf{x})} p(\mathbf{u} | C_j) d\mathbf{u}, \end{aligned}$$

where for mapping $f : U \rightarrow V$ and set $B \subset V$, $f^{-1}(B)$ represents the inverse image of B : $f^{-1}(B) = \{x \in U \mid f(x) \in B\}$, and also $\Pr(C_j) = \Pr\{Y = j\} = \Pr(Y^{-1}(\{j\}))$.

References

1. Duda, R.O., Hart, P.E., Stork, D.G. (2000). Chapter 2, Part 1. In *Pattern classification*. 2nd edn. New York: Wiley.

2. Fukunaga, K. (1990). Nonparametric classification and error estimation. In *Introduction to statistical pattern recognition*. 2nd edn. (pp. 313–321). San Diego: Academic Press.
3. Bishop, C. (2006). Decision theory. In *Pattern recognition and machine learning* (pp. 39–40). New York: Springer Science+Business Media.
4. Hastie, T., Tibshirani, R., Friedman, J. (2009). *Optimism of the error rate (in the elements of statistical learning)*, 2nd edn., (pp. 228–230). New York: Springer Science+Business Media.
5. Stone, M. (1974). Cross-validated choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, 36(2), 111–147.
6. Efron, B., & Tibshirani, R. (1993). *An introduction to the bootstrap*. New York: Springer Science+Business Media.
7. Demyanov, S., Bailey, J., Ramamohanarao, K., Leckie, C. (2012). AIC and BIC based approaches for SVM parameter value estimation with RBF kernels. *Journal of Machine Learning Research*, 25, 97–112.
8. Vapnik, V. (1998). The structural risk minimization principle. In Vapnik, V. (Ed.) *Statistical learning theory*. 1st edn. (pp. 219–268). New York: Wiley.
9. Guyon, I., Vapnik, V., Boser, B., Bottou, L., Solla, S.A. (1991). Structural risk minimization for character minimization. *Proceedings of Advances in Neural Information Processing Systems*, 4(NIPS 1991), 471–479.
10. Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6), 716–723.
11. Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2), 461–464.
12. Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of Machine Learning Research*, 11, 3571–3594.
13. Bouchard, G., & Celleux, G. (2006). Selection of generative models in classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4), 544–554.
14. Biem, A. (2003). A model selection criterion for classification: application to HMM topology optimization. In *Proceedings of IEEE seventh international conference on document analysis and recognition*.
15. Guo, Y., & Greiner, R. (2005). Discriminative model selection for belief net structures. In *Proceedings of AAAI twentieth national conference on artificial intelligence (AAAI-05)* (pp. 770–776).
16. Juang, B.-H., & Katagiri, S. (1992). Discriminative learning for minimum error classification. *IEEE Transactions on Signal Processing*, 30(12), 3043–3054.
17. Watanabe, H., Ohashi, T., Katagiri, S., Ohsaki, M., Matsuda, S., Kashioka, H. (2014). Robust and efficient pattern classification using large geometric margin minimum classification error training. *Journal of Signal Processing Systems, Springer*, 74(3), 297–310.
18. Ohashi, T., Watanabe, H., Tokuno, J., Katagiri, S., Ohsaki, M., Matsuda, S., Kashioka, H. (2012). Increasing virtual samples through loss smoothness determination in large geometric margin minimum classification error training. In *Proceedings of 2012 international conference on acoustics, speech and signal processing (ICASSP 2012)* (pp. 2081–2084).
19. Ha, D., Maes, J., Tomotoshi, Y., Melle, C., Watanabe, H., Katagiri, S., Ohsaki, M. (2017). A class boundary selection criterion for classification. In *Proceedings of IPSJ Kansai-Branch Convention 2017*, G-11.
20. Ha, D., Maes, J., Tomotoshi, Y., Watanabe, H., Katagiri, S., Ohsaki, M. (2018). A classification-uncertainty-based criterion for classification boundary. *IEICE Technical Report*, 117(442, PRMU2017-166), 121–126.
21. Ha, D., Delattre, E., Tomotoshi, Y., Senda, M., Watanabe, H., Katagiri, S., Ohsaki, M. (2018). Optimal classifier model status selection using Bayes boundary uncertainty. In *Proceedings of 2018 IEEE international workshop in machine learning for signal processing*.
22. Chapelle, O., & Vapnik, V. (2000). Model selection for support vector machines. *Advances in Neural Information Processing Systems*, 12.
23. von Luxburg, U., Bousquet, O., Scholkopf, B. (2004). Compression approach to support vector model selection. *Journal of Machine Learning Research*, 5, 293–323.
24. Demyanov, S., Bailey, J., Ramamohanarao, K., Leckie, C. (2012). AIC and BIC based approaches for SVM parameter value estimation with RBF kernels, (Vol. 25).
25. Bishop, C. (2006). Combining models. In Bishop, C. (Ed.) *Pattern recognition and machine learning* (p. 666).
26. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357.
27. Burges, C. (1998). A tutorial for support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 121–167.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.