INAUGURAL – DISSERTATION

ZUR
ERLANGUNG DER DOKTORWÜRDE
DER
NATURWISSENSCHAFTLICH–MATHEMATISCHEN–GESAMTFAKULTÄT
DER
RUPRECHT–KARLS–UNIVERSITÄT
HEIDELBERG

vorgelegt von
Dipl.-Math. oec. Jens Röder
aus Hermeskeil

Tag der mündlichen Prüfung: 24.01.2013

Active Learning:
New Approaches,
and Industrial Applications

Gutachter: Prof. Dr. Fred A. Hamprecht

Prof. Dr. Artur Andrzejak

Abstract

Active learning is one form of supervised machine learning. In supervised learning, a set of labeled samples is passed to a learning algorithm for training a classifier. However, labeling large amounts of training samples can be costly and error-prone. Active learning deals with the development of algorithms that interactively select a subset of the available unlabeled samples for labeling, and aims at minimizing the labeling effort while maintaining classification performance.

The key challenge for the development of so-called active learning strategies is the balance between exploitation and exploration: On the one hand, the estimated decision boundary needs to be refined in feature space regions where it has already been established, while, on the other hand, the feature space needs to be scanned carefully for unexpected class distributions. In this thesis, two approaches to active learning are presented that consider these two aspects in a novel way.

In order to lay the foundations for the first one, it is proposed to express the uncertainty in class prediction of a classifier at a test point in terms of a second-order distribution. The mean of this distribution corresponds to the common estimate of the posterior class probabilities and thus is related to the distance of the test point to the decision boundary, whereas the spread of the distribution indicates the degree of exploration in the corresponding region of feature space. This allows for the evaluation of the utility of labeling a yet unlabeled point with respect to classifier improvement in a principled way and leads to a completely novel approach to active learning. The proposed strategy is then implemented and evaluated based on kernel density classification.

The generic active learning strategy can be combined with any other classifier, but it performs best if the derived second-order distributions are sufficiently good approximations to the sampling distribution. Although second-order distributions for random forests are derived in this thesis, they do not approximate sufficiently well the sampling distribution and mainly allow only for the relative comparison of prediction uncertainty between test points. In order to combine the state of the art classification performance of random forests with the principal ideas of the first active learning approach, a related second approach for random forests is derived. It is, in addition, tailored to the demands in industrial optical inspection: bag-wise labeling with weak labels and strongly imbalanced classes. Moreover, an outlier detection scheme based on random forests is derived that is used by the proposed active learning algorithm.

Finally, a new computational scheme for Gaussian process classification is presented. It is compared to two standard methods in geostatistics, both with respect to theoreti-

cal consistency and practical performance. The method evolved as a by-product when considering using Gaussian process models for active learning.

Zusammenfassung

Aktives Lernen ist eine Form von überwachtem maschinellen Lernen. Beim überwachten Lernen wird eine Menge von gelabelten Beispielen an einen Lernalgorithmus übergeben, um einen Klassifikator zu trainieren. Das Labeln von großen Mengen an Trainingsdaten kann allerdings kostspielig und fehleranfällig sein. Aktives Lernen beschäftigt sich mit der Entwicklung von Algorithmen, die interaktiv eine Teilmenge der vorhandenen ungelabelten Beispiele für das Labeln auswählen, und zielt darauf ab, den Labelaufwand bei gleichzeitiger Erhaltung der Klassifikationsleistung zu minimieren.

Der Schlüssel zur Entwicklung von Aktiv-Lern-Strategien liegt in der Balance zwischen “Exploitation” und “Exploration”: Einerseits sollte die geschätzte Entscheidungsgrenze in den Regionen des Merkmalsraums verfeinert werden, wo sie bereits errichtet worden ist, andererseits sollte der Merkmalsraum sorgfältig nach unerwarteten Klassenverteilungen abgesucht werden. In dieser Arbeit werden zwei Ansätze zum aktiven Lernen vorgestellt, die diese beiden Gesichtspunkte auf neue Weise berücksichtigen.

Um die Grundlagen für den ersten Ansatz zu legen, wird zunächst vorgeschlagen, die Unsicherheit bzgl. der Klassenvorhersage eines Klassifikators an einem Testpunkt mit Hilfe einer Wahrscheinlichkeitsverteilung zweiter Ordnung auszudrücken. Der Mittelwert dieser Verteilung entspricht der bekannten Schätzung der posterioren Klassenwahrscheinlichkeiten und steht deshalb in Beziehung zur Entfernung des Punktes von der Entscheidungsgrenze, wohingegen die Streuung der Verteilung den Grad an Exploration der entsprechenden Region im Merkmalsraum anzeigt. Dies erlaubt eine Auswertung der Nützlichkeit des Labelns eines bisher ungelabelten Punktes in Bezug auf eine mögliche Verbesserung des Klassifikators auf grundlegende Weise und führt zu einem völlig neuen Ansatz für das aktive Lernen. Die vorgeschlagene Strategie wird schließlich basierend auf Kerndichteklassifikation umgesetzt und evaluiert.

Die generische Strategie kann mit jedem anderen Klassifikator kombiniert werden, aber sie ist am leistungsfähigsten, wenn die hergeleiteten Wahrscheinlichkeitsverteilungen zweiter Ordnung hinreichend gute Approximationen an die Stichprobenverteilung sind. Obwohl Verteilungen zweiter Ordnung für Zufallswälder (“random forests”) in dieser Arbeit hergeleitet werden, approximieren sie nicht hinreichend gut die Stichprobenverteilung und erlauben daher vor allem lediglich einen relativen Vergleich der Vorhersageunsicherheit zwischen Testpunkten. Um die anerkannt gute Klassifikationsleistung von Zufallswäldern mit den Grundideen des ersten Aktiv-Lern-Ansatzes zu verbinden, wird deshalb ein verwandter zweiter Ansatz für Zusatzwälder hergeleitet. Dieser ist zusätzlich auf die Anforderungen der industriellen Qualitätskontrolle zuge-

schnitten: bündelweises Labeln mit schwachen Labels und stark unbalancierte Klassen. Außerdem wird ein Verfahren zur Ausreißer-Erkennung basierend auf Zufallswäldern hergeleitet, das von dem vorgeschlagenen Aktiv-Lern-Algorithmus benutzt wird.

Abschließend wird ein neues Verfahren zur Klassifikation mit Gauß'schen Prozessen vorgestellt. Es wird mit zwei Standardmethoden aus der Geostatistik in Bezug auf das zugrunde liegende Modell und die Klassifikationsleistung verglichen. Die Methode entstand als Nebenprodukt bei der Überlegung, Gauß-Prozess-Modelle für aktives Lernen zu nutzen.

Acknowledgements

This work would not have been possible without the contributions of many people.

First, I would like to thank my doctoral advisor Prof. Dr. Fred Hamprecht from the Heidelberg Collaboratory for Image Processing (HCI) at the University of Heidelberg for his support and encouragement over the last years. This thesis would look much different without his commitment.

Further, I gratefully acknowledge financial support of this thesis by the Robert Bosch GmbH in Hildesheim. I had the opportunity to spend three interesting years in Schwieberdingen and Hildesheim and to gain knowledge in industrial optical inspection and video-based driving assistance. In particular, I would like to thank Dr. Wolfgang Niehsen, Jochen Wingbermühle and Walter Happold for giving me all the freedom I could wish for to pursue my research interests. Thanks also go to my industrial supervisors Prof. Dr. Michael Hißmann and Dr. Christian Perwass.

I'm also indebted to Dr. Boaz Nadler from the Weizmann Institute of Science in Rehovot, Israel, and Dr. Raimon Tolosana-Delgado from the Maritime Engineering Laboratory (LIM) in Barcelona, Spain, for their support and fruitful collaboration.

My time at Bosch would have been less enjoyable without my colleagues and fellow doctoral students. When work was fun, it was due to them. Many, many thanks—also for strong support and helpful scientific discussions—go to Andreas Walstra, Matthias Wieler, Patrick Sauer, Dr. Oliver Lange, Andreas Pfeil, Dr. Thomas Hörnlein, Dr. Florian Haug, Dr. Andreas Grützmann, Dr. Thomas Geiler, Holger Modler, Dr. Matthias Gebhard and Dr. Jan Warzelhan. I thank Andreas Pfeil also for carefully reading a draft version of this thesis.

I want to express my gratitude to all members of the image processing group at HCI. Special thanks go to Dr. Michael Hanselmann, Dr. Nikos Gianniotis, PD Dr. Ullrich Köthe, Dr. Bernhard Renard and Dr. Christoph Sommer for fruitful discussions and Rahul Nair and PD Dr. Ullrich Köthe for providing me with different C++ implementations of random forests.

Last but certainly not least, I would like to thank my mother for her support throughout my life. Sincere thanks also go to Steffi for her patience over the last years.

By three methods we may learn wisdom:
First, by reflection, which is noblest;
Second, by imitation, which is easiest;
and third by experience, which is the bitterest.

(Confucius)

Contents

1 Introduction

1.1 Scope of this Thesis

When searching for “Active Learning” in Google Scholar[1] on 4th November 2012, the first hit was a book entitled “Active Learning: Creating Excitement in the Classroom” [21]. Readers who expect this thesis to be about an educational technique, i.e. *human* learning, may be disappointed. However, this thesis *does* revolve around learning, *machine* learning to be more specific.[2]

Machine learning deals with the development of algorithms that allow computers to learn patterns from training data. Instead of simply memorizing the samples, a learning algorithm is supposed to recognize patterns and to generalize from them. This allows a machine learning system to analyze the data or to take decisions for unseen samples. The long list of possible applications includes such different areas as automated medical analysis [36, 98], speech recognition [92], handwritten character recognition [115] or industrial optical inspection [185]. As an example, in the latter application, sample images of production parts with and without defects may be provided. If the learning algorithm generalizes perfectly, images of new parts are then correctly classified as “intact” or “defective”.

As a rule of thumb, the more training data there is, the better the generalization ability of a learning algorithm. Unfortunately, in many applications, the creation of these samples requires the assignment of labels to the data by a human labeler. This can be time-consuming, expensive and/or error-prone. In the defect detection example, the images taken from production parts need to be labeled as “intact” or “defective”. Thousands or even millions of sample images can be taken during the production process. It is *not* a good idea to select a random subset of this data for labeling: If the manufacturer is quality-oriented, the overall majority of the parts is intact and looks very similar to each other. Most of these samples are thus not very interesting to the learning algorithm. As will be discussed in detail in Chapters 4 and 6, important are those parts with defects that look different from previously observed defects (found by “exploring” the space of possible images) and parts that are at the border between defective and intact (found by “exploiting” the unlabeled data at the border). The paradigm that aims at identifying the

[1] `http://scholar.google.de/`

[2] Note that there are attempts to explain the nature of human learning with machine learning approaches, see e.g. [173], [194], [128] and the references therein.

important unlabeled samples and thus at reducing the labeling effort as much as possible while maintaining the generalization ability of a learning algorithm is called “active learning” (AL).

This thesis focuses on the development of new approaches to active learning, in particular on balancing the tradeoff between exploration and exploitation.

1.2 The Scope in More Detail

At the highest level, machine learning algorithms can be divided into supervised and unsupervised algorithms. In *unsupervised* problems, real-world objects are represented by a feature vector only, i.e. a point in feature space. Typical tasks include the identification of groups of similar data points (clustering, see e.g. [95]), the detection of points that are distinct from the rest of the data (outlier detection, see e.g. [83]) or the estimation of the distribution the data points have been drawn from (density estimation, see e.g. [159]). In a typical scenario of *supervised* learning, the learning algorithm is given a set of so-called *training* samples, each consisting of a feature vector and an outcome measurement. The task is then to predict the outcome measurement of yet unseen objects represented by their feature vector. The task is called regression if the outcome is quantitative; it is called classification, if the outcome is one of a finite number of discrete categories, also called classes.

Active learning can be regarded as a variant of supervised learning. Whereas the task still is to predict the outcome measurement of unseen samples, the outcome measurement of most or even all of the training samples is unknown prior to the active learning process.[3] During this process, the learning algorithm is allowed to sequentially query the outcome measurement for some of the training samples. The algorithm that governs the selection process (aiming at selecting those samples that are most beneficial for the generalization ability of the learning algorithm) is called active learning strategy. Note that, as most of the training data miss their output measurement prior to and in early stages of the active learning process, also unsupervised learning techniques play a crucial role for the development of active learning strategies.

In this thesis, we concentrate on the development of *active learning* strategies for *classification*. Both concepts are introduced thoroughly in Sections 2.1 and 2.2, respectively.

1.3 Outline and Main Contributions

In **Chapter 2**, we introduce the basic concepts of classification and active learning.

[3] The scenario described here is called *pool-based* active learning and is introduced in detail in Section 2.2.1. Other, less common active learning scenarios will briefly be reviewed in Section 2.2.

As briefly explained in the previous section, the primary goal of classification is the prediction of the class label of yet unseen samples. Many learning algorithms additionally estimate the *probability* of an object belonging to a certain class. Although the latter is the most common indicator for the confidence in a prediction of class membership, it is still insufficient in many real-world applications. The main reason is that the estimated quantities may be highly inaccurate if the corresponding test samples lie in a feature space region that contains very few labeled training samples only. Instead, in addition to estimates of posterior class probabilities, some measure of *confidence* for these estimates is required as well. In **Chapter 3** of this thesis, we propose to express the confidence in estimates of posterior class probabilities in terms of second-order distributions. Applying a Bayesian approach, we first derive such distributional estimates for ε-nearest neighbors and then introduce "confidence k-nearest neighbors" and "confidence random forests" (CRF). We also investigate some of their finite sample and asymptotic properties in the limit of many labeled data.

Although the distributional estimates derived in Chapter 3 provide much more information than simple posterior class probability estimates, they do not provide an approximation of the true sample distribution, but allow only for a model-based relative comparison of the uncertainty in the prediction of posterior class probabilities. In **Chapter 4**, we derive distributional estimates for kernel density classification that indeed approximate the sample distribution. This allows for the development of a novel active learning strategy that trades off exploitation and exploration in a principled way. Loosely speaking, the distance to the decision boundary is encoded in the mean of the distributional estimate, whereas the degree of exploration in a certain region of the features space is indicated by the spread of the distribution.

Although the active learning strategy presented in Chapter 4 performs very well relative to other strategies, its absolute performance is limited by the properties of the employed classifier. Generative classifiers (like kernel density classification) generally perform worse than discriminative methods, in particular in high-dimensional feature spaces [20, chap. 1]. Therefore, the active learning strategy for defect detection presented in **Chapter 6** is based on the discriminative, state of the art random forest classifier. In order to combine this classifier with the basic ideas of the active learning approach presented in Chapter 4, we develop a related active learning strategy that does not require second-order distributions that approximate the true sample distribution. Instead, an extension of standard random forest can be used for the implementation. In addition, the strategy is tailored to the demands in industrial optical inspection: bagwise weak labels and strongly imbalanced classes. An important part of the active learning strategy, an outlier detection algorithm for random forests, is derived prior to the strategy in **Chapter 5**.

For a first implementation of the AL strategy presented in Chapter 6, indicator kriging had been used, a classification method based on Gaussian processes. But it turned out to be computationally more demanding and to perform worse than the random forest classifier. However, as a by-product, a new computational scheme for classification based on a doubly stochastic Gaussian process model has been derived. In contrast to previous inference methods for the same model, the estimation is analytical up to a final step where numerical integration is needed. This is presented in **Chapter 7**. In addition, the method and its underlying model is compared to two variants of indicator kriging—standard methods in geostatistics—with respect to theoretical consistency and practical performance.

As always when doing research, all effort raises more questions than it answers. The thesis concludes with a discussion on achievements and open problems in **Chapter 8**.

2 Preliminaries

2.1 Classification

2.1.1 Basic Terminology

In this section, we briefly present the necessary classification terminology. We start with an introduction to the underlying mathematical concepts.

Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space. Further, let $(\mathcal{X} \times \mathcal{Y}, \mathcal{B})$ be a measurable space and let

$$(X, Y) : \left\{ \begin{array}{ccc} \Omega & \to & \mathcal{X} \times \mathcal{Y} \\ \omega & \mapsto & (x, y) \end{array} \right.$$

be a random vector. The set $\mathcal{X} \subseteq \mathbb{R}^d$ is called feature space and the components $X^{(1)}, \ldots, X^{(d)}$ of X can be either continuous or discrete. The random variable Y is discrete, where $\mathcal{Y}$ is a *finite* set of classes. We denote by $\mathbf{P}$ the probability measure that is induced on $\mathcal{X} \times \mathcal{Y}$ by (X, Y), i.e.

$$\mathbf{P}(B) := \mathbf{P}^{(X,Y)}(B) := \mathbb{P}((X, Y)^{-1}(B)), \quad B \in \mathcal{B}$$

As is common in the machine learning literature, we denote the density of (X, Y) simply by $p(x, y)$, not distinguishing between Lebesgue and counting measure in the notation. The conditional probability $p(y|x) := p(Y = y|X = x)$ is called the posterior class probability of class y at point x, $p(x|y) := p(X = x|Y = y)$ is called the class density of class y (at point x).

A function

$$h : \left\{ \begin{array}{ccc} \mathcal{X} & \to & \mathcal{Y} \\ x & \mapsto & y \end{array} \right. \tag{2.1}$$

that assigns a class label to each feature vector is called a *classifier*. The quality of a classifier is usually assessed by some global risk functional [20, chap. 1]. The simplest among them is the *error rate* $\mathbf{P}(h(X) \neq Y)$, i.e. the probability of a wrong class assignment. The complementary probability $\mathbf{P}(h(X) = Y)$ is called *accuracy*. In practice, where the distribution of (X, Y) is usually not known, the error rate or other risk functionals cannot be calculated but need to be estimated [81, chap. 7]. This is briefly discussed in Section 2.1.2.

The central question in practice is how to obtain a good classifier. Obviously, we need some information about the distribution of (X, Y), i.e. any kind of realizations from (X, Y) or from its marginals. In the standard scenario of supervised classification, we are given a set $\mathcal{T} = \{(x_1, y_1), \dots, (x_n, y_n)\}$ of independent realizations of the random vector (X, Y), called training set or training data. The training set is then fed into a learning algorithm which outputs a classifier h. A discussion on properties of individual learning algorithms goes far beyond the scope of this introduction; the reader is referred to standard textbooks [81, 20, 53, 127]. We only mention here that some classifiers not only return a crisp class assignment as defined in Eq. (2.1), but also provide an estimate $\hat{p}(y|x)$ of the posterior class probability. In this thesis, we employ the ε-nearest neighbors classifier (Chapter 3), the k-nearest neighbors classifier (Chapter 3), random forests (Chapters 3, 5 and 6), the kernel density classifier (Chapter 4) and Gaussian process classification (Chapter 7). These classifiers are shortly introduced at the same place to make this thesis and its individual chapters self-contained.

2.1.2 Measuring Classification Performance

If the distribution of (X, Y) is not known, the error rate or other risk functionals for the assessment of classifier performance cannot be calculated but need to be estimated. The simplest way is to split up $\mathcal{T}$ into two sets, usually at a ratio of 2 to 1. The first one is used for training the classifier and the second one, called test set, is used for performance estimation.

However, if n is small, the quality of the classifier may suffer a lot from the reduction of the actual number of training samples. This motivates the application of cross-validation (CV). There, the training set is divided into k folds, where k is usually set to 5 or 10; $k - 1$ folds are used for training, one for testing. The latter is repeated k times such that each fold is used for testing once. Finally, the classifier is trained anew using all samples in $\mathcal{T}$. Its performance can be estimated by the mean of the k performance estimates for the individual test folds. Note that this estimate is conservative since the final classifier is trained with more samples than the classifiers for CV.

Test set and CV are the two techniques used in this thesis. A more thorough discussion on estimating classification performance can e.g. be found in [81, chap. 7] and the references therein.

2.2 Active Learning

In supervised learning, it is assumed that there is a completely labeled training set $\mathcal{T}$ available. The larger $\mathcal{T}$ is, the better is the generalization ability of a learning algorithm. While realizations of X can often be easily obtained in large quantities in practice, the corresponding class labels usually need to be provided by a human annotator. Active learning strategies aim at reducing this labeling effort while still achieving a high accuracy. The idea is to query labels only for those samples that are most important for classification performance.

As the reader can imagine, there are a lot of possible active learning scenarios and settings. The most important ones are presented in the rest of this section.

2.2.1 Pool-Based Active Learning

The setting considered in this thesis, which is also the most common, is called pool-based active learning. It is assumed that a small (possibly empty) set $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{l}$ of labeled data and a large pool $\mathcal{U} = \{x_i\}_{i=l+1}^{l+u}$ of unlabeled data is available prior to the active learning process. The feature vectors $\{x_i\}_{i=1}^{l+u}$ are assumed to be independent realizations of X, which is particularly important if an active learning strategy includes the estimation of the density $p(x)$. In contrast, it is not necessary to assume that the *labeled* samples are strictly independent; this assumption would be violated anyway during the active learning process: It is the key idea of active learning to create a *biased* training set.

As illustrated in Fig. 2.1, the strategies usually iterate between two steps:

1. Evaluate the training utility value (TUV) [86] for each $x \in U$.

2. Query the label y_* of $x_* = \arg\max_{x \in U} TUV(x)$, add (x_*, y_*) to $\mathcal{L}$ and remove x_* from $\mathcal{U}$.

A detailed overview of pool-based AL strategies is given in Chapter 4 when motivating the use of distributional estimates of posterior class probabilities for active learning. As an example, we here mention uncertainty sampling, a simple but commonly used strategy [105]. In the binary case with $\mathcal{Y} = \{0, 1\}$ and 0-1 loss [81, chap. 2], given a classifier that returns an estimate $\hat{p}(Y = 1|x)$ for posterior class probabilities, a label is queried for that point $x^* \in \mathcal{U}$ whose posterior prediction is closest to 0.5, i.e.

$$TUV(x) = 0.5 - |\hat{p}(Y = 1|x) - 0.5| \in [0, 0.5] \tag{2.2}$$

Note that in pool-based AL, all strictly monotonic transformations of a TUV lead to equivalent definitions.

A variant of the above scheme is batch mode AL [75], [84], where several instances are chosen in each iteration. This speeds up the AL process at the cost of lower AL performance due to possible “overlapping” information of the labels queried at the same time. We will apply batch mode AL in Chapter 6 in the context of defect detection.

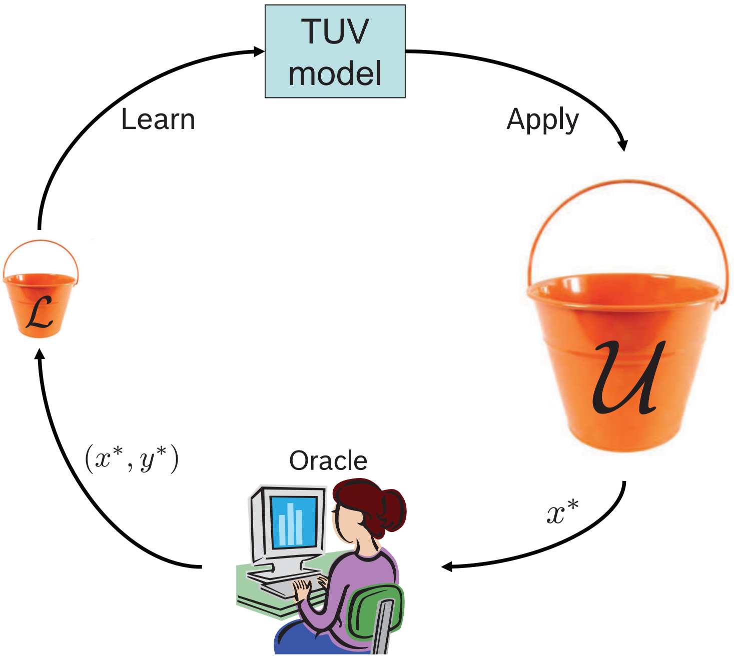


Figure 2.1: Active learning cycle in pool-based active learning. In each iteration, a TUV model is learned from the currently labeled training set $\mathcal{L}$ (possibly using information from $\mathcal{U}$). Then, applying this model to the elements in $\mathcal{U}$, the most valuable point x^* is identified and passed to an oracle, usually a human annotator. The latter provides the label for x^* and (x^*, y^*) is added to $\mathcal{L}$. Note that this figure is inspired by and similar to one in [163].

2.2.2 Other Active Learning Paradigms

Pool-based AL is the paradigm considered most often in the literature, but it is—of course—not the only one. In *stream*-based AL, the unlabeled data is not known prior to active learning. Instead, unlabeled data points are drawn sequentially from X and in each iteration it needs to be decided whether to discard the sample or whether to query a label for it and thus to add it to $\mathcal{L}$. In [43], several different classifiers are trained ("query by committee") and a label is queried based on a biased random decision taking into account the degree of committee agreement at the corresponding sample point. The same authors show in [8] that using two committee members only and simply querying a label for those points at which they disagree is sufficient to achieve a significant reduction in annotation costs.

Another scenario called "membership query synthesis" includes the possibility of requesting labels for any point in feature space [6] (see [39] for an example in the context of regression). Although this corresponds to an infinite pool of unlabeled data $\mathcal{U}$, the additional freedom has some drawbacks. First, since the unlabeled data is not drawn i.i.d. from X, many of the generated samples may be located in low (or even zero) density regions of feature space and thus their labeling may contribute not much to the improvement of classification performance. Second, membership query synthesis is not feasible for many applications. As an example, consider a blood test on a certain disease, i.e. we want to classify a person as healthy or not based on some blood measurements. Then, it will be difficult to find a person that has exactly those blood values a label is queried for. Third, even if it is possible to create examples de novo, the human annotator may not be able to provide a label for a certain sample if he cannot interpret it. The latter has e.g. been reported in the context of handwritten character recognition [14].

Note the relation between membership query synthesis and "Design of Experiments" (DoE, see e.g. [22], [41], [155]), where the latter is a statistical umbrella term that refers to methods that somehow control the information gathering process by querying output values for samples created de novo. However, DoE methods do usually not aim at learning a regression function or a classifier, but e.g. at finding some optimal parameter configuration, i.e. at identifying the point in feature space with the largest or smallest (expected) response. For many DoE methods, e.g. factorial design experiments, feature space is assumed to be discrete and to have a few dimensions only [22]. In ANOVA, several labels are queried at each point in feature space, and a statistical test on whether the responses at different points differ significantly is performed.

2.2.3 Does Active Learning Work?

The short answer is "yes", both theoretically and empirically.

From a theoretical point of view, it has been shown in [10] and [62] in a stream-based scenario that the labeling effort indeed can be substantially reduced using AL. In both references, it is shown that exponentially fewer labeled samples may be sufficient to achieve the same error rate as with passive learning, i.e. sampling the training instances randomly from (X, Y). Unfortunately, the derivations rely on assumptions that are usually not satisfied in practice or that can at least not be verified for a particular real-world data set.

However, the empirical evidence that the labeling effort indeed can be reduced using AL seems overwhelming (although some caution is called for due to publication bias); see this thesis and all AL references therein, in particular in Chapters 4 and 6. In [164], 17 different AL strategies are compared to each other using 8 different data sets showing the benefit from AL. Moreover, we are not aware of any systematic investigation showing e.g. that the application of a commonly used AL strategy does not lead to a reduction of labeling effort.[1]

[1] Note that—of course—an AL strategy may perform worse than passive learning on individual data sets using a specific classifier, see e.g. [156], [75].

2.3 Related Learning Paradigms

Active learning is not the only approach considered in the literature to save labeling time. Two others, namely semi-supervised learning (SSL) and active class selection, are briefly presented in the following.

2.3.1 Semi-Supervised Learning

In (pool-based) active learning, unlabeled data is available in great quantities and the learning algorithm is allowed to query labels for selected instances. In contrast, in SSL [33, 192], the subset of labeled instances is initially given and a classifier is trained using both the labeled *and* the unlabeled examples. The unlabeled data can help by assuming that two points have the same label with high probability if there is a path between them that passes only through regions of relatively high density of the feature distribution ("cluster assumption") [160]. Active learning and semi-supervised learning can be combined by learning the TUV model or the final classifier at the end of the AL process using both labeled and unlabeled data [132, 193].

2.3.2 Active Class Selection

In some applications, it is not possible to obtain feature data without knowing the corresponding label. As an example, consider the "artificial nose" in [114] that is supposed to automatically distinguish between different chemical vapors of interest. There, training data is created by first generating the vapor and then passing it over an array of sensors. "Active class selection" is the AL analogue that addresses this scenario. Instead of querying labels at certain positions in feature space, realizations from a certain class are queried. In [114], five different query strategies are compared and a substantial reduction of labeling effort (compared to sampling equally from all classes) is achieved if samples are queried for the most "unstable" classes. Stability is defined by number of class predictions that change after having added the samples from the last iteration. The latter is calculated by cross-validating the training set of the current and the previous iteration.

3 Distributional Uncertainty Estimates

As briefly explained before in Section 2.1, in the standard framework of classification, the goal is to construct a classifier with small risk, e.g. measured by the error rate. In many real world applications, simple predictions of class labels are typically insufficient and point estimates for posterior class probabilities are used as an indication for the confidence in a label prediction. In this chapter, we show that these quantities may be highly inaccurate for test samples not well represented by the training set. Instead, some measure of confidence in these point estimates is required as well. We propose a Bayesian framework to derive such confidence measures in terms of second-order distributions over the posterior probabilities. We apply our approach to several popular classifiers, including ε-NN, k-NN and random forests. The utility of our approach, which unifies classification and outlier detection, is illustrated on real world datasets from machine vision (road sign recognition) and from imaging mass spectrometry.

3.1 Introduction

One of the basic principles in statistical inference is to "never give an estimator without giving a confidence set"[1]. This principle extends far beyond the field of statistics and applies to essentially all of scientific research. In analytical chemistry, for example, it is well recognized that "an analytical result is not complete until a statement about its uncertainty is formulated" [136]. Whereas many works developed confidence or prediction intervals for multivariate regression and calibration (see [112, 58, 121, 174] and references therein), perhaps surprisingly, uncertainty estimates in classification have thus far received much less attention. Most classifiers provide either just a predicted class label, or at best a point estimate of posterior class probabilities.

Yet, deriving a measure of uncertainty is at least as important in classification as it is in multivariate regression: First, an important practical issue is to have an indication of where the classifier may be at error. Beyond samples near the classification boundary, test samples with high uncertainty are also likely candidates for classification errors. Moreover, letting a classifier make class predictions for new observations

[1] Quote from the preface of Wasserman's textbook [181].

poorly represented by the training set may lead to drastic adverse consequences in several practical applications such as medical diagnostics, industrial process control, and automated safety systems. The ability to measure the uncertainty of predicted posterior class probabilities also plays a key role in active learning methods [163], as it can help assess in a principled way the degree of exploration in a given region of feature space (see Chapter 4). Moreover, it may also help to detect that the underlying distribution of the test data differs from that of training data or becomes different over time (population drift) [78].

In this chapter, we aim to bridge this gap regarding uncertainty estimates in classification. We develop a Bayesian strategy that provides sample-specific "uncertainty" or "confidence" estimates for several popular local averaging classifiers, including ε-nearest neighbor (ε-NN), k-nearest neighbor (k-NN) and random forests (RF). Note that we are not interested in the trivial *first-order* or *ambiguity uncertainty*, encountered wherever different classes overlap significantly in feature space. In such cases, when the point estimates for posterior probabilities of the different classes are all far from one, the classifier might announce "doubt" instead of predicting a class label (e.g. [150, chap. 2]). In contrast, the focus and main contribution of this chapter is the derivation of an uncertainty measure over these point estimates, via a full *second-order distribution*, that is, a probability distribution over the unknown posterior probabilities.

In classification, there is hence a hierarchy of possible outputs at new test points: at the simplest level, a mere class label prediction; at an intermediate level, a point estimate of the posterior class probabilities, allowing for an ambiguity reject; and at the most refined level, a second-order distribution over the posterior probabilities, allowing to extract different kinds of confidence or uncertainty statements. In the following sections, we present a Bayesian framework for deriving such distributional estimates, and illustrate their importance in a variety of applications.

3.1.1 Related Work

In the theoretical framework of classification, given a training set and a loss function for incorrect predictions, the task is to construct a classifier with small generalization risk, typically the overall test error rate. Hence, most classifiers report at best point estimates of these quantities. Furthermore, there is a non-trivial bias-variance decomposition for classification [63], which in particular implies that a classifier may (nearly) achieve the optimal Bayes error rate and yet have biased point estimates of posterior probabilities. The fact that the overall error of a classifier depends only on its estimates of posterior probabilities has perhaps concealed the importance of uncertainty measures for these quantities, and may explain why uncertainty estimates have received relatively little attention thus far.

Nonetheless, the need for uncertainty measures has been recognized. Several authors developed methods to highlight ambiguous predictions, i.e. test samples where no single class has a dominant posterior probability [38, 74, 76]. These methods, however, rely on point estimates, the intermediate level of the hierarchy discussed above.

Other works focused on the detection of outliers, though typically separately from the actual classifier [46, 52]. Closer to our work is [44] which uses the classifier itself to find outlying samples. This task is relatively straightforward for generative classifiers which learn a model for the joint density $p(x, y)$. At ambiguous samples none of the estimated class densities dominates the others, whereas at outliers the overall estimated density is very low [139, 101, 119].

Confidence intervals for classification were derived for nonparametric generative regression techniques, notably indicator kriging [90, 175] and for parametric discriminative techniques, most notably logistic regression [124].

3.1.2 Distributional Estimates for Local Averaging Classifiers

In this chapter, in contrast, we consider uncertainty estimates for various popular nonparametric local averaging discriminative classifiers, and present a Bayesian framework to compute sample-specific second-order distributions for their posterior class probabilities. Our proposed framework thus allows for a unified treatment of both classification and outlier detection.

After introducing notations and the problem setup in Section 3.2, in Section 3.3 we first illustrate our approach for one of the simplest possible local classifiers: The ε-nearest neighbor classifier (ε-NN). Given a suitable prior, standard Bayesian methods yield an approximate second-order distribution for the posterior class probabilities. If the selected prior is conjugate to the binomial distribution, our Bayesian scheme is analytically tractable with simple explicit formulas. From a theoretical perspective, assuming that the posterior class probabilities are approximately constant inside ε-balls, a natural result is that the larger the number of training samples inside an ε-ball, the more concentrated the second-order distribution around the true posterior class probability becomes. Furthermore, in an appropriate joint limit, as $\varepsilon \to 0$ and as the number of training samples tends to infinity, the distribution converges to the true posterior (Section 3.7.2.2).

Next, in Section 3.4 we consider the more popular and typically more accurate k-nearest neighbor classifier (k-NN). Unfortunately, the above method to construct second-order distributions does not carry over to the k-NN classifier. The reason is that for any query point, the k-NN classifier always uses the labels of the k nearest neighbors to a query point, regardless of their distance. To develop distributional estimates in this setting, in a first step, we augment the training set with samples from an auxiliary class with uniform density (referred to as "data-driven confidence k-NN"). The resulting multi-class classification problem with these auxiliary points effectively puts an upper limit on the radius of the prediction neighborhood, thus leading, as desired, to second-order distributions with large spread in areas poorly represented by the original training set (Section 3.4.1). In a second step, we develop a model where the samples from the auxiliary class no longer need to be drawn explicitly, but are considered implicitly in a probabilistic way (referred to as "probabilistic confidence k-NN", Section 3.4.2). The resulting classifier and its uncertainty estimate depend on a data-driven kernel function which weights the nearest neighbors by their distance from the query point while maintaining the local adaptivity of k-NN.

In Section 3.5, the arguments developed previously for the ε-NN and k-NN classifiers culminate in an approximate second-order distributional estimate for random forests, a discriminative ensemble classifier [24]. Random forests (RF) have enjoyed a soaring popularity in recent years, and for good reasons: in a broad array of applications [66, 69, 140], it is one of the best-performing classifiers [11, 42], with very little tuning required. To derive the desired second-order distributions we present a small modification of the original algorithm, which we denote "confidence random forests" (CRF) (Section 3.5.3). While sufficient conditions for consistency can be stated (Section 3.7.2.4), in general these are not met by CRF, and it remains unknown whether they can be weakened. We note that this theoretical gap is to be expected, since even the consistency of the standard version of RF is yet to be proven [18].

After deriving distributional estimates for the various local averaging methods, we present their practical application in Section 3.8. First, we illustrate their working and potential usefulness using simple toy data. Then, we show the importance of uncertainty estimates in two real-life applications: a digital pathology example using mass spectrometric multi-spectral images and a speed sign recognition task from machine vision.

3.2 Problem Setup and Notation

To simplify the exposition, in this chapter we focus on the case of binary classification. We note that the proposed approach easily generalizes to multi-class problems.

Let (X, Y) be a random vector with probability density $p(x, y)$, where $x \in \mathcal{X} \subseteq \mathbb{R}^d$ is a feature vector and $y \in \mathcal{Y} = \{1, 2\}$ is its class label. Further, let $\pi(x) := p(Y = 2|X = x)$ denote the posterior probability of class 2 at x. Finally, let the training set $\mathcal{T}_n = \{(x_i, y_i)\}_{i=1}^n$ be composed of n i.i.d. samples from (X, Y). Throughout this chapter (and this thesis), the letter $\mathcal{P}$ denotes a distribution, whereas p denotes a probability density and $\mathbf{P}$ denotes the probability of a specific event.

As discussed in the introduction of this chapter, at the simplest level, the goal of training a classifier on a set $\mathcal{T}_n$ is to estimate the labels of new (test) samples so as to minimize some global risk functional. At the next level, if detection of test samples with potentially ambiguous predictions is important, a point estimate $\hat{\pi}(x)$ of the true posterior class probability $\pi(x)$ is additionally desired. At the most refined level, when the *reliability* of these point estimates is of essence, the goal is to obtain an uncertainty estimate, preferably from a full second-order distribution or *distributional estimate* with a distribution function

$$F_x(q) = \hat{\mathbf{P}}(\pi(x) \leq q|\mathcal{T}_n), \quad q \in [0,1] \tag{3.1}$$

Deriving such a distributional estimate is the central goal of this chapter. In the following three sections, we develop Bayesian inference models for three popular local averaging classifiers, ε-NN, k-NN and RF, that take into account the inevitable uncertainty of posterior probability estimates incurred when training data are scarce in the vicinity of a query point.

Before describing the technical details, let us first illustrate these concepts by the one-dimensional toy example shown in Fig. 3.1. The true class densities in the upper panel have the following distributions:

$$\begin{aligned}&\mathcal{P}(X|Y=1) = \mathcal{N}(0,3^2)\\&\mathcal{P}(X|Y=2) = 0.5\cdot\mathcal{N}(-3,2^2) + 0.5\cdot\mathcal{N}(6,2^2)\end{aligned}$$

where $\mathcal{N}(\mu,\sigma^2)$ denotes the Gaussian distribution with mean μ and variance σ^2. Setting $p(Y=1) = p(Y=2) = 1/2$ results in the posterior class probabilities $\pi(x)$ depicted in the same panel in red. A training set with $n = 1000$ samples is depicted as short vertical bars in the lower panel of Fig. 3.1, together with the resulting distributional estimates obtained from CRF (described in detail in Section 3.5).

As expected, the resulting distributional estimates are relatively narrow at test points well represented by the training data. In contrast, these distributional estimates become broader as the query point moves away from the training set, and eventually tend to a user-defined prior distribution for queries very far from the training data. These broad distributional estimates indicate a (very) low confidence in the posterior prediction, advising the user to exercise extreme caution in interpreting the predicted labels at such test samples.

To further illustrate this important point, consider a test sample at $x = 20$, whose true posterior probability is $\pi(20) = p(Y=2\,|X=20) = 0.07$. Since in the interval $x > 10$ the given training set contains five samples from class 2 and none from class 1, the posterior estimate $\hat{\pi}(20)$ of standard random forests would be very close to one. In contrast, our modified confidence RF not only gives a conservative point estimate of about $1/2$, but also endows it with a very broad distribution, indicating a high uncertainty in the predicted posterior at this test point.

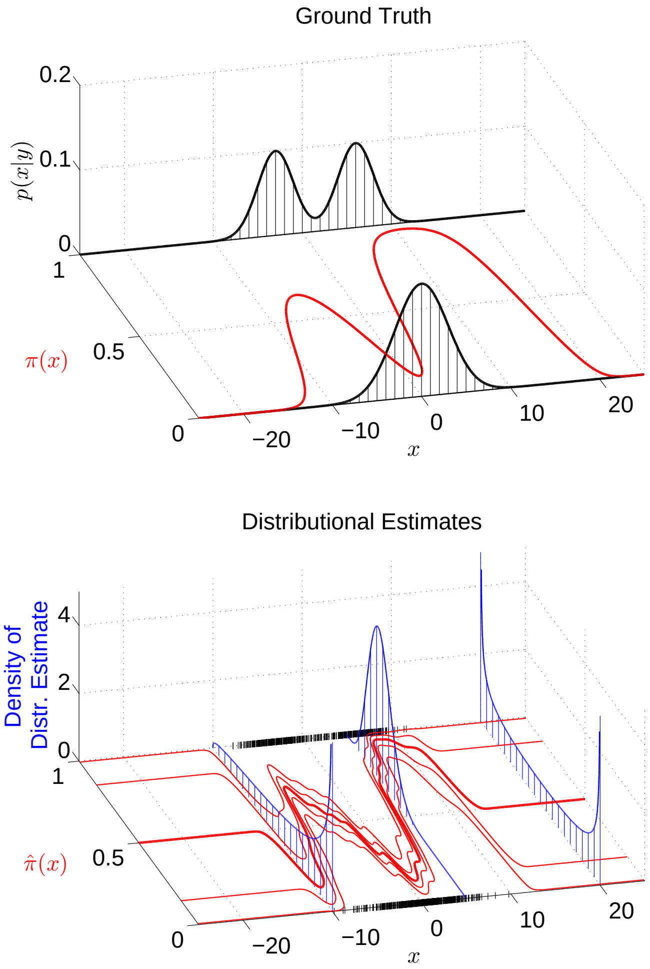


Figure 3.1: [Best viewed in color] Distributional estimates for a 1-D toy problem. **Top Panel:** The two black curves are the unknown densities of classes 1 and 2, respectively. The red curve is the posterior probability $\pi(x)$ of class 2. **Bottom Panel:** A training set $\mathcal{T}$ with $n = 1000$ i.i.d. samples from the ground truth is depicted as short vertical bars. Instead of computing a mere point estimate for $\pi(x)$, in this work we estimate second-order distributions for $\pi(x)$ (see Sections 3.3-3.5). The 0.05, 0.25, 0.5, 0.75 and 0.95-quantiles of the distributional estimates from confidence RF are plotted using red lines. Three examples of the corresponding probability density functions are plotted in blue at $x = -10$, $x = 5$ and $x = 20$. The fewer training samples there are in the vicinity of a query point, the higher the uncertainty in posterior prediction and the larger the variance of the corresponding second-order distribution. For query points far from all training data the distributional estimate tends to the prior, a $Beta(1/2, 1/2)$ distribution in our case.

3.3 Distributional Estimates for ε-Nearest Neighbors

The ε-nearest neighbor classifier estimates the posterior class probabilities at a query point by counting the number of training samples of the different classes within radius ε of the point of interest. To derive a second-order distribution for the posteriors, we employ the standard Bayesian inference model for the probability of success of the binomial distribution.

In more detail, let $\|\cdot\|$ denote the Euclidean norm on $\mathbb{R}^d$, and let $B_\varepsilon(x)$ be the closed ball around x with radius ε. Further, denote by

$$n_{i,x} = \left|\{(x', y') \in \mathcal{T}_n : x' \in B_\varepsilon(x), y' = i\}\right|, \quad i = 1, 2$$

the number of training samples of class i inside $B_\varepsilon(x)$.

If $\pi(x)$ exhibits some degree of smoothness, for example if $\pi(x)$ is continuous with Lipschitz constant L, for sufficiently small ε it follows that $\pi(x)$ is approximately constant inside $B_\varepsilon(x)$. Therefore, the labels of the training samples inside $B_\varepsilon(x)$ can be modeled as independent Bernoulli realizations with the same success parameter $\pi(x)$. Hence $n_{2,x}$ is binomially distributed with the observed parameters $(n_{1,x} + n_{2,x})$ as the number of experiments, and the unknown $\pi(x)$ as the probability of success.

A second-order distribution for $\pi(x)$ is obtained by applying a standard Bayesian inference model for the success parameter of the binomial distribution (see e.g. [20, chap. 2]). Specifically, let p_B denote the probability density function of a prior second-order distribution $\mathcal{P}_B$ for the success parameter $\pi(x)$. Then, the second-order distribution for $\pi(x)$ at a query point x with $n_{1,x}$ and $n_{2,x}$ labeled samples of classes 1 and 2 respectively, is given by

$$p(\pi(x)|n_{1,x}, n_{2,x}) \propto \mathbf{P}(n_{1,x}, n_{2,x}|n_{1,x} + n_{2,x}, \pi(x))\, p_B(\pi(x)) \tag{3.2}$$

where $\mathcal{P}(n_{1,x}, n_{2,x}|n_{1,x} + n_{2,x}, \pi(x)) = Binom(n_{1,x} + n_{2,x}, \pi(x))$. Note that if there are no labeled samples in the ε-ball of a test point x ($n_{1,x} = n_{2,x} = 0$), our second-order distribution for $\pi(x)$ is just the prior $\mathcal{P}_B$. The prior second-order distribution $\mathcal{P}_B$ thus represents our best guess for the posterior probability $\pi(x)$ in the absence of (nearby) observations. If we have no bias favoring one class over the other, the prior should be symmetric around the value $\pi(x) = 1/2$. Natural choices are the uniform distribution or Jeffreys' uninformative prior [88]. For a binomial, the latter is a Beta distribution with parameters $(1/2, 1/2)$, see Fig. 3.2. While its bimodal appearance may seem puzzling at first sight, it actually makes sense in the classification context: In most regions of feature space, different classes usually have little overlap. Little overlap between classes, in turn, matches the prior belief that $\pi(x)$ is either close to 0 or close

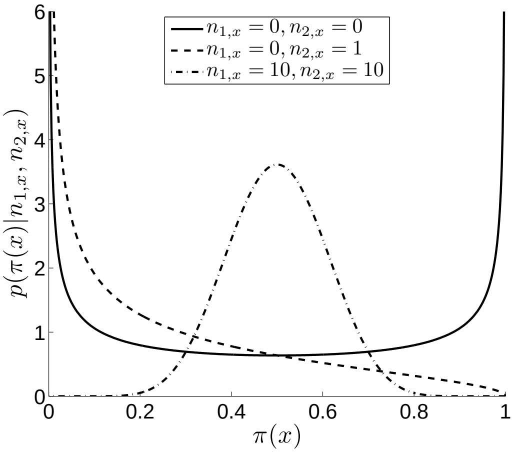


Figure 3.2: Examples of probability densities of the Beta distribution with parameters $1/2 + n_{2,x}$ and $1/2+n_{1,x}$. The solid curve is Jeffreys' prior (with parameters 1/2 and 1/2). The other two curves illustrate the resulting posterior distributions of $\pi(x)$ given $(n_{1,x}, n_{2,x}) = (0, 1)$ or $(n_{1,x}, n_{2,x}) = (10, 10)$ samples of classes 1 and 2 inside $B_\varepsilon(x)$. Note that the variance of the distribution decreases as the number of observations inside $B_\varepsilon(x)$ increases.

to 1 anywhere in feature space—precisely what Jeffreys' prior happens to embody.

Choosing Jeffrey's prior $\mathcal{P}_B(\pi(x)) = Beta(1/2, 1/2)$ as discussed above leads to (see e.g. [20, chap. 2])

$$\mathcal{P}(\pi(x)|n_{1,x}, n_{2,x}) = Beta(1/2 + n_{2,x}, 1/2 + n_{1,x}) \tag{3.3}$$

whose corresponding (Bayesian) point estimate for the posterior probability is

$$\hat{\pi}(x) = \frac{n_{2,x} + 1/2}{n_{1,x} + 1/2 + n_{2,x} + 1/2}. \tag{3.4}$$

One theoretical question is how accurate the distributional estimates of Eq. (3.3) are. A second theoretical question is the asymptotic consistency of the point estimate in Eq. (3.4) in the limit of large training data. We consider these issues in some detail in Section 3.7.2. At this point, we note that if indeed the posterior $\pi(x)$ is constant inside $B_\varepsilon(x)$, then as discussed in [29], confidence intervals extracted from Jeffreys' prior using Eq. (3.3) are quite accurate, even when the number of samples is relatively small. In addition, at least for the case of ε-NN, one is not necessarily restricted to a Bayesian approach, and frequentist confidence intervals, such as those described in [4], would be equally applicable. Finally, note that by the Bernstein-von Mises theorem, asymptotically frequentist confidence intervals and Bayesian credible sets are very close to each other.

To summarize, Eq. (3.3) is the resulting second-order distribution for the ε-NN classifier. As expected, in high density regions with $n_{1,x} + n_{2,x} \gg 1$, the resulting distributional estimates are quite narrow, indicating a high confidence in the estimated posteriors. In contrast, the distributional estimates become broader as $n_{1,x} + n_{2,x}$ decreases, and in the extreme case of no labeled points, they revert to the prior distribution for $\pi(x)$.

3.4 Distributional Estimates for k-Nearest Neighbors

The k-NN classifier estimates the posterior class probabilities at a query point by counting the number of training samples of the different classes within the k nearest neighbors of the point of interest. Whereas ε-NN relies on a fixed neighborhood radius, the k-NN classifier has an adaptive neighborhood size that adjusts to local density. While this typically leads to improved classification performance, by implicitly implementing some kind of bias-variance trade-off, it no longer allows us to use the simple recipe from the previous section: As k-NN uses exactly k training points to answer each query, the above Bayesian inference or standard frequentist confidence intervals would hence insinuate comparable uncertainty estimates at all query points, regardless of their distances from labeled data.

To derive sensible uncertainty estimates for the k-NN classifier, we propose to introduce auxiliary reference data and generalize the classification problem and related distributional estimates from a two-class binomial setting to a multi-class multinomial setting. In more detail, let $(x_{(1)}, y_{(1)}), \ldots, (x_{(n)}, y_{(n)})$ be the ordered sequence of the training data with respect to their distance to the point x, i.e. $x_{(j)}$ is the j-th nearest neighbor of x. Further, let $B_k(x) = \{x' \in \mathcal{X} : \|x' - x\| \leq \|x_{(k)} - x\|\}$ be the closed ball around the point x with radius equal to the distance to its k-th nearest neighbor.

To construct second-order distributions for the k-NN classifier, we build on an idea previously employed for a variety of unsupervised problems, including outlier detection, density estimation and one-class learning. These unsupervised problems can all be transformed into supervised classification problems by generating artificial samples from a reference distribution and learning the dichotomy between observed and artificial data using a discriminative classifier (see e.g. [81, chap. 14], [170], [87], [172], [25]). As an example, consider the problem of estimating the density $p(x)$. To this end, let $\mathcal{R}$ be a hyper-rectangle that covers the feature vectors of the training set, i.e. $\{x_1, \ldots, x_n\} \subset \mathcal{R}$. We draw n_0 samples $\mathcal{S}_0 = \{x_{n+1}, \ldots, x_{n+n_0}\}$ from the uniform distribution on $\mathcal{R}$, label them as class 0, label the original samples as class "12" and create the new training set $\mathcal{T}_d = \{(x_1, 12), \ldots, (x_n, 12), (x_{n+1}, 0), \ldots, (x_{n+n_0}, 0)\}$. Let

$p_0 = 1/\mathrm{vol}(\mathcal{R})$ denote the constant density of class 0. Then,

$$\begin{aligned} p(Y=12|x) &= \frac{p(x)p(Y=12)}{p(x)p(Y=12)+p_0 p(Y=0)} \\ \iff \qquad p(x) &= p_0 \frac{p(Y=0)}{p(Y=12)} \cdot \frac{p(Y=12|x)}{p(Y=0|x)} \end{aligned} \tag{3.5}$$

The terms on the right hand side of Eq. (3.5) are either known or can be estimated by a classifier—trained on $\mathcal{T}_d$—that predicts posterior class probabilities.

The main difference from the above scheme is that in our setting, we add auxiliary samples not to unlabeled or single-class data but to the training set of a supervised learning problem. To motivate our approach and to lay the ground for derivations in Sections 3.4.2 and 3.5, in a first step (Section 3.4.1), we explicitly generate the auxiliary samples. As the auxiliary samples are random, in a second step we consider the average of the resulting confidence estimates over infinitely many realizations. For the k-NN classifier, this can be done analytically by probabilistic arguments (Section 3.4.2).

3.4.1 Data-Driven Confidence k-NN

As above, let $\mathcal{R}$ be a hyper-rectangle that covers the feature vectors of the training set. We draw n_0 samples $\mathcal{S}_0 = \{x_{n+1}, \ldots, x_{n+n_0}\}$ from the uniform distribution on $\mathcal{R}$, label them as class 0 and append them to the original training set $\mathcal{T}_n$. We denote the new (three-class) training set by $\tilde{\mathcal{T}}_{n'} = \mathcal{T}_n \cup \{(x_{n+1}, 0), \ldots, (x_{n'}, 0)\}$ where $n' = n + n_0$. Further, we denote by $B_k^0(x)$ the ball around x whose radius is given by the distance to the k-th nearest neighbor in the augmented training set $\tilde{\mathcal{T}}_{n'}$. Note that by definition $B_k^0(x) \subseteq B_k(x)$.

We use the number of artificial samples in $B_k^0(x)$ as an indicator for the posterior prediction uncertainty at a point x. For a test point x located in a high density region, the radius of $B_k^0(x)$ is only little (or not at all) smaller than that of $B_k(x)$, with very few or no representatives from the reference class 0. In contrast, for a query point in a low density region, the majority or even all of its nearest neighbors are from the reference class $\mathcal{S}_0$. In this case, the radius of $B_k^0(x)$ may be substantially smaller than the radius of $B_k(x)$.

In analogy to the previous section, to obtain a distributional estimate for the posterior class probabilities, we now apply standard Bayesian inference, only this time for the multinomial rather than the binomial distribution. We introduce the following notation: For $i \in \{0, 1, 2\}$, let $p_{i|x}$ be the true unknown posterior probability of class i in the *three*-class problem. It can be easily shown that the posterior of the original two-class problem can be calculated from those of the three-class problem by

$$\pi(x) = \frac{p_{2|x}}{p_{1|x} + p_{2|x}} \tag{3.6}$$

Further, let $n_{i,x}$ be the number of labels of class i in $B_k^0(x)$.[2] Then,

$$p(p_{0|x}, p_{1|x}, p_{2|x} | n_{0,x}, n_{1,x}, n_{2,x}) \propto \mathbf{P}(n_{0,x}, n_{1,x}, n_{2,x} | p_{0|x}, p_{1|x}, p_{2|x}) p_B(p_{0|x}, p_{1|x}, p_{2|x})$$

where p_B is the density of a prior distribution for $(p_{0|x}, p_{1|x}, p_{2|x})$. In particular, choosing Jeffreys' uninformative prior for multinomial inference [88], i.e.

$$\mathcal{P}_B(\pi(x)) = Dir(1/2, 1/2, 1/2),$$

where $Dir(\alpha_0, \alpha_1, \alpha_2)$ is the Dirichlet distribution with parameters α_0, α_1 and α_2, yields (e.g. [20, chap. 2])

$$\mathcal{P}(p_{0|x}, p_{1|x}, p_{2|x} | n_{0,x}, n_{1,x}, n_{2,x}) = Dir(1/2 + n_{0,x}, 1/2 + n_{1,x}, 1/2 + n_{2,x}). \tag{3.7}$$

Using Proposition 3.11 from the appendix of this chapter in Section 3.10, it readily follows from Eqs. (3.6) and (3.7) for the posterior of the original two-class problem that

$$\mathcal{P}(\pi(x) | n_{0,x}, n_{1,x}, n_{2,x}) = Beta(1/2 + n_{2,x}, 1/2 + n_{1,x}). \tag{3.8}$$

Note the similarity between Eqs. (3.8) and (3.3). The only difference is the interpretation of $n_{i,x}$, which in the case of ε-NN is the number of labels of classes 1 and 2 inside $B_\varepsilon(x)$, whereas in the case of k-NN it is their number inside $B_k^0(x)$.

Similar to the case of ε-NN, if a test sample x is in a high density region, then most, if not all, of its neighbors are from the original labeled set, resulting in relatively narrow distributional estimates, and indicating a high confidence in the posterior estimate. In contrast, if the test sample is in a region of such low density that all its neighbors are from the reference class, the distributional estimate reverts to the prior. Finally, note that the regularization parameters n_0 and k play a role similar to the parameter ε in the ε-NN classifier. A method to set n_0 and k in order to obtain sensible distributional estimates is discussed in Section 3.5.

[2] In there are multiple training samples at the boundary of the ball $B_k^0(x)$, which leads to $n_x := \sum_{i=0}^{2} n_{i,x} > k$, we can simply choose at random $n_x - k$ of these points located on the boundary and ignore them.

3.4.2 Probabilistic Confidence k-NN

The distributional estimates in the previous section depend on the specific realization $\mathcal{S}_0$ of auxiliary samples. We now show that it is actually not necessary to explicitly generate the auxiliary data to obtain distributional estimates for the k-NN classifier. Instead, we can compute the *exact* probability that a training sample from $B_k(x)$ will be displaced from among the k nearest neighbors by samples from the auxiliary class, when averaging over infinitely many realizations of $\mathcal{S}_0$. The complementary probability that a training sample remains in $B_k^0(x)$ can be used to weight the sample accordingly.

As before, let $x_{(j)}$ be the j-th NN of a query point x. First of all, note that $x_{(j)}, j = 1, \ldots, k$, is in $B_k^0(x)$ if and only if there are at most $k - j$ points in $\mathcal{S}_0$ whose distance to x is smaller than $\|x - x_{(j)}\|$. Let $V(r_j)$ be the volume of the d-dimensional ball with radius $\|x - x_{(j)}\|$. Then,

$$\mathbf{P}\left(x_{(j)} \in B_k^0(x)\right) = \sum_{i=0}^{k-j} \binom{n_0}{i} \left(\frac{V(r_j)}{\mathrm{vol}(\mathcal{R})}\right)^i \left(\frac{\mathrm{vol}(\mathcal{R}) - V(r_j)}{\mathrm{vol}(\mathcal{R})}\right)^{n_0 - i} \tag{3.9}$$

Defining $\rho := n_0/\mathrm{vol}(\mathcal{R})$, expression (3.9) equals

$$\begin{aligned} &\sum_{i=0}^{k-j} \frac{n_0!}{i!(n_0-i)!} \left(\frac{\rho V(r_j)}{n_0}\right)^i \left(1 - \frac{\rho V(r_j)}{n_0}\right)^{n_0-i} \\ =&\sum_{i=0}^{k-j} \left(1 - \frac{\rho V(r_j)}{n_0}\right)^{n_0-i} \frac{(\rho V(r_j))^i}{i!} \frac{n_0!}{n_0^i (n_0-i)!} \end{aligned} \tag{3.10}$$

Since we no longer explicitly generate any auxiliary samples, there is no need to restrict the uniform distribution to a finite hyper-rectangle $\mathcal{R}$. Instead, we can assume that the auxiliary class is distributed throughout the feature space with uniform density ρ. So, fixing ρ in expression (3.10) while letting n_0 (and thus $\mathrm{vol}(\mathcal{R})$) go to infinity yields

$$\mathbf{P}(x_{(j)} \in B_k^0(x)) = e^{-\rho V(r_j)} \sum_{i=0}^{k-j} \frac{(\rho V(r_j))^i}{i!} =: A_{(j)} \tag{3.11}$$

Note that $A_{(j)} = 0$ for $j > k$. Now, instead of applying the Bayesian inference scheme proposed above by merely counting the number of labels of classes 1 and 2 in $B_k^0(x)$, we can *weight* each training sample $(x_{(j)}, y_{(j)})$ in $B_k(x)$ by its probability $A_{(j)}$ of being in $B_k^0(x)$. This yields

$$\mathcal{P}(\pi(x)|\mathcal{T}_n) = Beta\left(1/2 + \sum_{j:y_{(j)}=2} A_{(j)},\ 1/2 + \sum_{j:y_{(j)}=1} A_{(j)}\right) \tag{3.12}$$

This amounts to the use of fractional votes, or equivalently to the use of a kernel function. In contrast to standard Nadaraya-Watson type estimators where the kernel width is fixed, here, the kernel function depends not only on the distance of $x_{(j)}$ to x, but also on the distances to x of all training samples that are closer to x than $x_{(j)}$, thus maintaining the local adaptivity of the k-NN approach.

To conclude, Eq. (3.12) is our proposed second-order distribution for the posterior $\pi(x)$ of the probabilistic k-NN classifier. It consists of a weighted sum of votes of its original k-NN, where the weight of an observation is its probability of still being one of the k-nearest neighbors after having added auxiliary data from the reference class. This probability is an analytic expression that can be computed, via Eq. (3.11), without explicitly generating the reference data. Obviously, the regularization parameters ρ and k play a role similar to n_0 and k in data-driven confidence k-NN. For a method to set these parameters, see Section 3.5.

3.5 Confidence Random Forests

In this section we derive distributional estimates for random forests (RF). For the chapter to be self-contained, we first briefly describe the original RF classifier. Next, we present a small modification of the underlying tree construction that provides distributional estimates for posterior class probabilities for each single tree, obtained by explicitly adding reference data. The derivations for probabilistic confidence k-NN in the previous section do not easily carry over to tree-based classifiers. Instead, we combine many individual distributional estimates, each obtained from a different tree with its own random realization of $\mathcal{S}_0$, into a single one for a forest ensemble.

3.5.1 Standard Random Forests

The RF classifier [24] is an ensemble learner consisting of M decision trees. To build an individual tree, a bootstrap sample is drawn from the training set and recursively divided until all leaf nodes contain instances of a single class only. For the split at a certain node, $dtry < d$ out of the d feature dimensions are randomly selected and the best axis-orthogonal split according to a purity measure (the Gini criterion) on these $dtry$ variables is used. An estimate for the posterior class probability at a test point x is obtained by passing it down all the trees and dividing the number of trees that vote for the respective class by M. The majority vote yields a class assignment.

3.5.2 A Distributional Estimate for a Single Tree

A single tree learned from a training set can equivalently be represented as a partition Π of feature space into disjoint cells which correspond to the different leaves of the tree. For any point x, we denote by $\Pi(x)$ the leaf cell containing x. To construct a distributional estimate for the posterior probability $\pi(x)$, a simple possibility is to consider all training samples in the same leaf as x. This idea is explored here; however, as discussed below, several issues need to be resolved beforehand.

First, we face the same problem as in the case of k-NN (Section 3.4): Learning a tree and simply counting the number of training samples in the cell $\Pi(x)$ does not consider the (potentially very large) distances of the training samples in $\Pi(x)$ to the query point x. Even if the number of training samples in a cell $\Pi(x)$ is large, the estimate for $\pi(x)$ may still be unreliable if the query point x is far from all of them. As in the case of k-NN, to overcome this problem we propose to add auxiliary data to the training set. As before, let $\mathcal{S}_0 = \{x_{n+1}, \dots, x_{n+n_0}\}$ be an i.i.d. sample from the uniform distribution on a hyper-rectangle $\mathcal{R}$ that covers the original training set, and let $\tilde{\mathcal{T}}_{n'} = \mathcal{T}_n \cup \{(x_{n+1}, 0), \dots, (x_{n'}, 0)\}$ be the augmented training set. Instead of constructing a tree with the original training set $\mathcal{T}_n$, we use the augmented set $\tilde{\mathcal{T}}_{n'}$ and train a three-class tree. As for k-NN, this decreases the “sphere of influence” of each training sample.

Next, recall that in the standard RF algorithm, tree nodes are partitioned until all leaves contain labels of a single class only. Obviously, having only pure cells yields neither sensible point estimates nor sensible distributional estimates for the posterior probability $\pi(x)$ from a single tree. To overcome this limitation, we propose to slightly modify the tree construction as follows: we require, in analogy to k-NN, that leaves contain no less and not many more than k training samples. More specifically, at each node we choose the best split only among those that result in children with at least k samples each. In particular, nodes with at least $2k$ samples are split, whereas nodes with less than $2k$ samples are not split, regardless of their class labels. Such “early stopping” rules have long been promoted for the regularization of decision trees (see [110, 120] and references therein).

With these modifications in effect, distributional estimates for posterior probabilities can be constructed in a fashion similar to the case of k-NN: Let x be a test point, $\Pi(x)$ the cell it belongs to, and let $n_{i,x}$ be the number of labeled samples in $\Pi(x)$ of class $i = 0, 1, 2$, respectively. Blindly applying the standard Bayesian scheme would lead to Eq. (3.8) for the distributional estimate for $\pi(x)$, whose corresponding point estimate is

$$\hat{\pi}(x) = \frac{n_{2,x} + 1/2}{n_{1,x} + 1/2 + n_{2,x} + 1/2}. \tag{3.13}$$

There is, however, one subtle difference between our setting and that of k-NN. Whereas in the case of k-NN, the total number of neighbors is always k ($n_x := n_{0,x} + n_{1,x} + n_{2,x} = k$), here the total number varies between k and $2k - 1$. Hence, even if two query points x_1, x_2 have the same values of $n_{1,x}, n_{2,x}$, the point estimates derived from Eq. (3.13) may still have different uncertainties depending on the value of $n_{0,x}$. To illustrate this point, suppose $k = 5$ and consider two query points $x_1 \neq x_2$, with $n_{1,x_1} = n_{1,x_2} = 1$, and $n_{2,x_1} = n_{2,x_2} = 4$, but $n_{0,x_1} = 0$ and $n_{0,x_2} = 4$. Obviously, the second query point lies in a lower density region. As its cell contains more auxiliary points, its uncertainty estimate should be somewhat larger.

To take into account the variable number (between k and $2k - 1$) of training samples inside $\Pi(x)$, we simply normalize the impact of each training sample in $\Pi(x)$ by the factor k/n_x. Applying this normalization and repeating the same calculations as above, the final distributional estimate of $\pi(x)$ from a single tree is given by

$$\mathcal{P}(\pi(x)|n_{0,x}, n_{1,x}, n_{2,x}) = Beta(1/2 + \tilde{n}_{2,x}, 1/2 + \tilde{n}_{1,x}) \tag{3.14}$$

where $n_{i,x}, i = 0, 1, 2$, is the number of training samples in $\Pi(x)$ with label i, and $\tilde{n}_{i,x} = n_{i,x} \cdot k/n_x$. Note that with this normalization in place, to minimize tree size, *pure* nodes containing more than $2k$ samples need not be split.

3.5.3 A Distributional Estimate for a Forest

As described above, the random forest classifier uses M decision trees for class prediction. Averaging over trees typically yields more robust and accurate predictions than those obtained from each of the individual trees. Moreover, using many trees opens here the possibility of drawing a different set of auxiliary samples for each tree and thus minimizes the influence of a particular realization of $\mathcal{S}_0$ on the final distributional estimate.

Let x be a query point and denote by $\Pi^{(m)}(x), m = 1, \ldots, M$, the cell of the m-th tree-based partition that contains x. Further, let $\tilde{n}_{i,x}^{(m)}$ be the normalized number of training samples with label i in $\Pi^{(m)}(x)$. A natural way of combining the numbers $\tilde{n}_{i,x}^{(m)}$ is to average over the contents of the random partitions $\Pi^{(m)}(x)$ of the M different trees. This gives

$$\mathcal{P}\left(\pi(x)\left|\{n_{0,x}^{(m)}, n_{1,x}^{(m)}, n_{2,x}^{(m)}\}_{m=1}^{M}\right.\right) = Beta\left(1/2 + \frac{1}{M}\sum_{m=1}^{M}\tilde{n}_{2,x}^{(m)},\ 1/2 + \frac{1}{M}\sum_{m=1}^{M}\tilde{n}_{1,x}^{(m)}\right) \tag{3.15}$$

The statistical interpretation of this approach is that, as in probabilistic confidence k-NN, each training sample in $\mathcal{T}_n$ is weighted by the fraction of trees for which it is in the same cell as the query point x. The difference is that the fractional votes are computed analytically for k-NN, whereas they are obtained by averaging over many trees for CRF. The working principle of CRF is illustrated in Fig. 3.1 with $n_0 = n = 1000$, $k = 30$ and $\mathcal{R} = [-15, 15]$.

Finally, we remark that [187] considered a similar tree-based ensemble model for estimating the confidence in posterior probabilities, albeit with different splitting rules for individual trees. The main difference is that in [187], the estimated confidence increases with the number of trees in the ensemble, leading to unrealistically small confidence intervals in large ensembles. Moreover, [187] does not consider the uncertainty arising from a poor representation of a test sample by the training data, and hence cannot detect outliers.

3.6 Selecting the Regularization Parameters

Both confidence k-NN and CRF have two parameters that need to be set. The first parameter is the number of neighbors k in k-NN, or analogously the minimum occupancy k of a leaf cell in CRF. For probabilistic confidence k-NN, the second parameter is the density ρ of auxiliary points. For data-driven confidence k-NN and CRF, the second parameter is the number of auxiliary samples n_0.[3] For simplicity, we only refer to n_0 in the following; analogous statements hold for ρ.

By and large, these parameters provide a bias-variance trade-off: increasing k and reducing n_0 leads to larger prediction neighborhoods, hence more bias and less variance. In high density regions where $np(x) \gg n_0/\mathrm{vol}(\mathcal{R})$, the size of a leaf mostly depends on k and is almost independent of n_0. Conversely, for fixed k, the size of a neighborhood in low density regions mainly depends on n_0.

An automated choice of these parameters is not obvious. One reason is that it is not possible to measure the quality of distributional estimates in the common setting where the validation set contains only labeled samples, rather than their true posterior class probabilities [72]. As an example, consider a test sample of class 2. Its likelihood given a distributional estimate is equal to the corresponding point estimate $\hat{\pi}(x)$. That is, a given test sample does not allow us to distinguish the quality of two second-order distributions so long as they have the same expectation value. Also, optimizing the parameters by cross-validation (CV) with respect to classification accuracy may yield $n_0 = 0$ if the training set is free from outliers. But setting $n_0 = 0$ does not allow us to flag outlying test samples, the identification of which is a central point of this work.

Instead, we propose a two step approach. In the first step, we set $n_0 = 0$ and determine the optimal value for k by CV. In a second step, with k fixed, one option for the k-NN classifier is to compute, in the training set, the mean d_k and standard deviation σ_k of the distance to the k-th NN, and set the density ρ such that $\rho = 1/Vol(B_{d_k+2\sigma_k})$. This ensures that for most training samples the probability of an auxiliary sample inside the k-NN ball is relatively small. A different option for the second step, again keeping

[3] Besides n_0, one also needs to set $\mathcal{R}$. Choosing it too large means that resources are wasted because many auxiliary points are used to redundantly shield off empty space. Choosing it too small may lead to insufficient protection against outliers. However, the rectangle $\mathcal{R}$ is not a parameter in a strict sense, but only needs to be “sufficiently” large. For fixed $\mathcal{R}$, the ratio $n_0/\mathrm{vol}(\mathcal{R})$ and thus n_0 needs to be chosen.

k fixed, is to determine the value of n_0 also via CV, by choosing the largest possible value of n_0 that does not harm prediction performance. “Harm” is measured here by the 1-standard error-rule (1-SE-rule), as proposed in [81, chap. 7.10] or [27, chap. 3.4] for model selection. This rule states that performance differences may be neglected if they are smaller than the standard error of the estimated performance. In our context, we thus choose the largest n_0 for which the estimated accuracy is not more than one standard error below the highest estimate for the accuracy for all n_0 used in CV. The procedure is demonstrated in Section 3.8.1 and used throughout the rest of the experimental section. In another experiment in Section 3.8.2, we show that there is a high correlation between the rank order of the estimated uncertainties for different choices of k and n_0. In simple words, the ability to detect outliers is very robust with respect to the exact parameter choice.

3.7 Theoretical Analysis

As briefly discussed in Section 3.3, there are two key theoretical questions associated with the distributional estimates derived in this chapter. The first is with respect to their accuracy, and the second is with respect to their asymptotic consistency (defined explicitly below). In this section we consider these questions in some detail.

3.7.1 Accuracy of Distributional Estimates for Finite Sample Size

The distributional estimates derived for ε-NN, k-NN and CRF are based on the assumption that the posterior $\pi(x)$ is constant inside $B_\varepsilon(x)$, $B_k^0(x)$ or $\{\Pi^{(m)}(x), m = 1, \ldots, M\}$, depending on the classifier employed. Here, we investigate the error incurred if this assumption is not satisfied. We first focus our discussion on ε-NN.

According to Eq. (3.2), the second order distribution for the posterior $\pi(x)$ at a point x with $n_{1,x}$ and $n_{2,x}$ labels of class 1 and 2, respectively, inside $B_\varepsilon(x)$ is

$$p(\pi(x)|n_{1,x}, n_{2,x}) \propto \mathbf{P}(n_{1,x}, n_{2,x}|n_{1,x} + n_{2,x}, \pi(x))p_B(\pi(x)) \tag{3.16}$$

where

$$\mathbf{P}(n_{1,x}, n_{2,x}|n_{1,x} + n_{2,x}, \pi(x)) = \binom{n_{1,x} + n_{2,x}}{n_{2,x}} \pi(x)^{n_{2,x}}(1 - \pi(x))^{n_{1,x}}.$$

Let Z be a random test sample from the same distribution as X and let Y be its label. In general, the probability that $Y = 2$ given that $Z \in B_\varepsilon(x)$ is not equal to $\pi(x)$. Rather, it is given by

$$\bar{\pi}_\varepsilon(x) = \mathbf{P}(Y = 2|Z \in B_\varepsilon(x)) = \mathbf{E}[\pi(Z)|Z \in B_\varepsilon(x)].$$

Thus, assuming a total number of $n_{1,x} + n_{2,x}$ training samples inside $B_\varepsilon(x)$, the probability of observing $n_{1,x}$ and $n_{2,x}$ labels of class 1 and 2, respectively, is given by

$$\mathbf{P}(n_{1,x}, n_{2,x} | n_{1,x} + n_{2,x}, \bar{\pi}_\varepsilon(x)) = \binom{n_{1,x} + n_{2,x}}{n_{2,x}} \bar{\pi}_\varepsilon(x)^{n_{2,x}} (1 - \bar{\pi}_\varepsilon(x))^{n_{1,x}}$$

Hence, Eq. (3.16) is in fact the standard Bayesian inference model for $\bar{\pi}_\varepsilon(x)$ and not for $\pi(x)$. If $\bar{\pi}_\varepsilon(x) = \pi(x)$, Eq. (3.16) becomes the exact Bayesian inference model for the success parameter of a binomial distribution. As discussed in [29], the extracted confidence intervals are quite accurate, even for a small number of samples. If $\bar{\pi}_\varepsilon(x) \neq \pi(x)$, the error we incur is small if $|\bar{\pi}_\varepsilon(x) - \pi(x)|$ is small. For example, if $\pi(x)$ is Lipschitz continuous with constant L, then $|\bar{\pi}_\varepsilon(x) - \pi(x)| \leq L\varepsilon$. It follows that the difference tends to 0 for $\varepsilon \to 0$.

The above considerations also apply to k-NN and CRF, where the size of the neighborhood considered depends on k and n_0. If $\pi(x)$ is equal to the average posterior in $B_k^0(x)$ or in $\{\Pi^{(m)}(x), m = 1, \ldots, M\}$, the inference model for $\pi(x)$ is exact. The error we incur increases with the absolute difference between $\pi(x)$ and the average posterior. Putting an upper bound on this difference is exactly the raison d'être of adding the artificial data.

3.7.2 Consistency of the Proposed Distributional Estimates

In this section, we investigate the consistency of the proposed *distributional* estimates for ε-NN, confidence k-NN and CRF. More specifically, we state sufficient conditions under which the proposed distributional estimates converge to the true posterior class probability $\pi(x)$ as the size of the training set increases to infinity. For the ε-NN and data-driven confidence k-NN classifiers, these conditions are easily satisfied by setting the respective parameters accordingly. For probabilistic confidence k-NN, we additionally need a very mild assumption on the distribution of X. For CRF, in contrast, it is still an open question if the suggested splitting rules imply consistency. In this respect we remark that even the consistency of the original RF is yet to be proven [18].

Note that the consistency of posterior *point* estimates for standard ε-NN and k-NN classifiers are proven in [49] and [47], respectively. Here, we additionally show that (i) the distributional estimates converge to the same limit as the point estimates, in particular that the distributional estimates converge to the degenerate distribution, and that (ii) adding the auxiliary data preserves consistency provided their number n_0 does not increase too fast with n. The proofs of the propositions stated in this section are given in the appendix of this chapter.

3.7.2.1 Preliminaries

In the following, let μ be the probability measure of X, i.e. $\mu(A) = \int_A p(x)dx$ for all measurable $A \subset \mathcal{X}$ if $p(x)$ is a Lebesgue density. The distribution of (X,Y) is completely determined by the pair (μ, π) (see [48, chap. 2]).

First, we recall the definition of point-wise convergence in probability. To this end, the posterior point estimate of a method is denoted by $\hat{\pi}_n(x)$ instead of $\hat{\pi}(x)$ to explicitly state the dependence on the size of the training data.

Definition 3.1. *The sequence $\{\hat{\pi}_n\}_n$ is called weakly consistent iff*

$$\lim_{n\to\infty} \hat{\pi}_n(x) = \pi(x) \; in\; probability\; for\; \mu\text{-}almost\; all\; x$$

where μ is the probability measure of X.

Definition 3.2. *The sequence $\{\hat{\pi}_n\}_n$ is called strongly L_1-consistent iff*

$$\lim_{n\to\infty} \mathbf{E}_X \left[|\hat{\pi}_n(X) - \pi(X)| \right] = 0 \; a.s.$$

Remark 3.3. *As pointed out in [27, chap. 12], the point-wise consistency in Definition 3.1 is equivalent to*

$$\lim_{n\to\infty} \mathbf{E}_{\mathcal{T}_n} \left[\mathbf{E}_X \left[|\hat{\pi}_n(X) - \pi(X)| \right] \right] = 0$$

This form of uniform consistency is called weak L_1-convergence.

Remark 3.4. *As the names suggest, strong L_1-convergence implies weak L_1-convergence.*

When stating sufficient conditions for the consistency of the distributional estimates for ε-NN, confidence k-NN and CRF, we refer to point-wise convergence in probability as defined in Definition 3.1. In the proofs in the appendix, we make use of the implication in Remark 3.4 and the equivalence in Remark 3.3.

In Proposition 3.9, which refers to the consistency of the distributional estimates of probabilistic confidence k-NN, we need the mild assumption that $\mathbf{E}[|X|] < \infty$, whereas in Propositions 3.7, 3.8 and 3.10, the sufficient conditions for consistency of the estimates are valid for *any* distribution of (X,Y). In particular, X is not required to have a density in any of the consistency proofs. This is reflected by the following definition.

Definition 3.5. *A method is called universally consistent if the sequences $\{\hat{\pi}_n\}_n$ obtained therefrom are consistent for all distributions of (X,Y).*

Finally, we state the following proposition for later use.

Proposition 3.6. *Let $\{a_{i,n}\}_n, i = 1, 2$, be two sequences of non-negative real valued random numbers and let $\{Z_n\}_n$ be a sequence of Beta distributed random variables with parameters $1/2 + \alpha_{1,n}$ and $1/2 + \alpha_{2,n}$. Assume that $\alpha_{1,n} + \alpha_{2,n} \xrightarrow{\mathcal{P}} \infty$ as $n \to \infty$ and that $\alpha_{1,n}/(\alpha_{1,n} + \alpha_{2,n}) \xrightarrow{\mathcal{P}} \alpha_1/(\alpha_1 + \alpha_2)$. Then,*

$$Z_n \xrightarrow{\mathcal{P}} \frac{\alpha_1}{\alpha_1 + \alpha_2}$$

3.7.2.2 Consistency of the Distributional Estimate for ε-NN Classification

We first state sufficient conditions for the universal consistency of the distributional estimate for ε-NN. On one hand, a required condition for consistency is that the radius of $B_\varepsilon(x)$ converges to zero as $n \to \infty$; this ensures that $\pi(x)$ is sufficiently smooth on $B_\varepsilon(x)$. On the other hand, this convergence needs to be sufficiently slow such that the number of samples in $B_\varepsilon(x)$ still increases to infinity.

Proposition 3.7. *Assume that*

$$\varepsilon(n) \to 0 \quad \text{and} \quad n\varepsilon(n)^d \to \infty \text{ as } n \to \infty \tag{3.17}$$

Then, the proposed distributional estimate in Eq. (3.3) *is consistent.*

3.7.2.3 Consistency of the Distributional Estimate for k-NN Classification

Now, we consider consistency of confidence k-NN. We start with a proposition regarding the data-driven version. As before for ε-NN, the number of training samples $k = k(n')$ in the neighborhood $B_k^0(x)$ of x has to increase to infinity as $n' \to \infty$. However, k should increase sufficiently slowly such that the radius of $B_k^0(x)$ still decreases to zero. Finally, n_0 should increase sufficiently slowly so that the fraction of samples from the original training set $\mathcal{T}_n$ within the k nearest neighbors does not converge to zero.

Proposition 3.8. *Assume that*

$$k(n') \to \infty \quad \text{and} \quad \frac{k}{n'} \to 0 \text{ as } n' \to \infty \tag{3.18}$$

and that

$$n_0/n = O(1) \tag{3.19}$$

Then, the proposed distributional estimate in Eq. (3.8) *is consistent.*

Remark. *Condition* (3.19) *implies that n, the number of samples in the original training set, increases to infinity as $n' \to \infty$.*

Next, we consider the consistency of probabilistic confidence k-NN. Here, under the additional assumption that $\mathbf{E}[|X|] < \infty$, we have the following proposition:

Proposition 3.9. *Assume that*

$$k(n) \to \infty \quad and \quad \frac{k}{n} \to 0 \; as \; n \to \infty \tag{3.20}$$

and that

$$\rho/n = O(1) \tag{3.21}$$

Further, assume that $\mathbf{E}[|X|] < \infty$. *Then, the proposed distributional estimate in Eq.* (3.12) *is consistent.*

3.7.2.4 Consistency of Confidence Random Forest

We first state sufficient conditions for the universal consistency of the distributional estimate based on a single tree. Then, we discuss the implications for the consistency of confidence RF. As discussed below, it is an open question whether confidence RF is universally consistent.

The sufficient conditions we derive for the universal consistency of the distributional estimate of a single tree are similar to those for confidence k-NN, i.e., loosely speaking, the number of samples in the neighborhood needs to increase to infinity while the neighborhood diameter needs to shrink to zero. However, k needs to increase faster than $\log(n')$ here and the definition of the neighborhood size is a little more involved.

Proposition 3.10. *Let* $\Pi_{n'}$ *be the partition obtained by growing a tree with the augmented training set* $\mathcal{T}'_{n'}$ *and with parameter* $k(n')$, *where*

$$\frac{k(n')}{\log n'} \to \infty \; as \; n' \to \infty \tag{3.22}$$

Assume further that

$$n_0/n = O(1) \tag{3.23}$$

and that the following "shrinking cell condition" is satisfied for every $\gamma > 0$ *and* $\delta \in (0,1)$*:*

$$\inf_{S:\mu(S)\geq 1-\delta} \mu\{x : diam(\Pi_{n'}(x) \cap S) > \gamma\} \to 0 \; a.s. \tag{3.24}$$

where $diam(A) = \sup_{x,y\in A} ||x-y||$. *Then, the proposed distributional estimate in Eq.* (3.14) *is consistent.*

Remark. *Condition* (3.24) *is a "shrinking cell condition". It basically demands that the probability mass of* X *contained in cells whose diameter is larger than* γ *decreases to 0. Taking the infimum over all sets* S *that contain a minimum mass of* X *allows* *for cells that are infinitely large as long as the mass outside a subset of this cell with diameter γ is sufficiently small.*

Proposition 3.10 states sufficient conditions for the universal consistency of the distributional estimate of a classifier based on a single tree. According to the splitting rules presented in Section 3.5.2, it is easy to ensure that condition (3.22) is satisfied by setting k accordingly. Unfortunately, the "shrinking cell condition" (3.24) is not satisfied for every distribution of (X, Y). As an example, assume that $x = (x^{(1)}, x^{(2)})^T \in \mathcal{X} = [0, 1]^2$, that $p(x|Y = 1) = 1$ and $p(x|Y = 2) = 1/2 + x^{(1)}$. Assuming equal priors, it follows that $p(Y = 2|x) = 1 - 2/(3 + 2x^{(1)})$, i.e. the posterior increases from $1/3$ to $3/5$ in the first dimension and is constant with respect to the second dimension. For sufficiently large n and $dtry = 2$, all splits are made orthogonal to the first dimension. Hence, although the volume of the cells shrinks if $k(n')$ is set appropriately, their diameter will not tend to zero but is equal to one. However, note that shrinking cells are a sufficient but not a necessary condition. For the distribution considered, the splits are made in such a way that the posterior probabilities are approximately constant in each cell. This is a desired property and yields consistent estimates if $k(n')$ satisfies Condition (3.22).

It is an open question if there are distributions for which the proposed distributional estimate is not consistent.[4] It may be necessary to adapt the splitting rules to enforce splits in alternate directions. Since confidence RF simply averages over the observations of different tree-based partitions, the universal consistency of the distributional estimates of confidence RF may follow from that of the individual trees. It is another open question if the application of bootstrapping, which violates the i.i.d. assumption of the training data, allows for consistency.

As a final remark, we note that while the original RF algorithm has shown empirically a very good classification performance, it has so far resisted a complete theoretical analysis, leading to minor or major modifications of the original algorithm in most theoretical publications on the subject. Given the choice between a more complete theoretical analysis and greater proximity to the original implementation, we opted for the latter.

3.8 Results

In this section, we first illustrate the effect and optimization of the design parameters (Section 3.8.1) and then show the usefulness of distributional estimates on two real world data sets, one from road sign recognition (Section 3.8.2) and a second from an imaging mass spectrometry (IMS) experiment (Section 3.8.3). In all experiments involving CRF, the number of trees M is set to 100 and $dtry = \sqrt{d}$ according to the rule

[4] Note that the two-dimensional example given in [48, chap. 20] on page 335 for a slightly different splitting rule does not apply here. The reason is that instead of fixing the number of splits allowed, we fix the approximate number of samples in each cell.

of thumb proposed by the inventor of RF [25].

3.8.1 The Parameters k, n_0 and ρ

In the following, we use a toy data example to investigate the role of the regularization parameters k, ρ and n_0. Further, we demonstrate their optimization as proposed in Section 3.6.

Consider a two-dimensional binary classification problem, where the class-conditionals $X|Y = 1$ and $X|Y = 2$ are normally distributed with unit covariance matrix and mean $(1,1)^T$ and $(-1,-1)^T$, respectively. A training set $\mathcal{T}_n$ with $p(Y = 1) = p(Y = 2) = 1/2$ and $n = 200$ is plotted in Fig. 3.3.

We first concentrate on probabilistic confidence k-NN. With ρ fixed, the larger k, the larger the “sphere of influence” of the original training samples. This obvious relation is illustrated in Fig. 3.3 by setting $\rho = 1$ and increasing k from 5 to 10. With k fixed, the larger ρ, the higher the probability that a point from the hypothetical reference data $\mathcal{S}_0$ is closer to a test sample x than one or several of the k nearest neighbors from $\mathcal{T}_n$. Hence, the variance of the distributional estimate increases and the mean will be more influenced by training set labels close to x. This is illustrated in Fig. 3.3 by increasing ρ from 1 to 2 while setting $k = 5$.

In the last row of Fig. 3.3, standard k-NN predictions for the posterior class probability $\pi(x)$ are compared to the point estimates (mean values of the Beta distribution) of probabilistic confidence k-NN ($\rho = 1$, $k = 5$). They are approximately equal where the feature space is covered by training data, but confidence k-NN predictions tend more to $1/2$ in the low density parts of feature space. Note that the low confidence in these predictions is indicated by a high variance of the corresponding Beta distribution.

Similar observations to those for probabilistic confidence k-NN can be made for CRF, where the parameter ρ is replaced by n_0. The corresponding results are shown in Fig. 3.4. The rectangle $\mathcal{R}$ is set to $[-10, 10]^2$.

In Section 3.6, we proposed an approach for optimizing k and n_0 (or ρ). We demonstrate the procedure based on the toy data set shown in the upper left panel of Fig. 3.4. We initially set $n_0 = 0$ ($\rho = 0$) and determine the optimal value for k with respect to accuracy by 5-fold cross-validation (CV). This yields $k = 3$ for CRF and $k = 13$ for probabilistic confidence k-NN. Then, we evaluate the accuracy of CRF (with $k = 3$) and confidence k-NN (with $k = 13$) for varying values of n_0 or ρ. This is shown in Fig. 3.5, together with the standard errors obtained from CV. According to the 1-SE-rule proposed in Section 3.6, we choose the largest n_0 (ρ) for which the estimated accuracy is not more than one standard error below the highest estimate for the accuracy. This yields $\rho = 10$ for probabilistic confidence k-NN and $n_0 = 700$ for CRF.

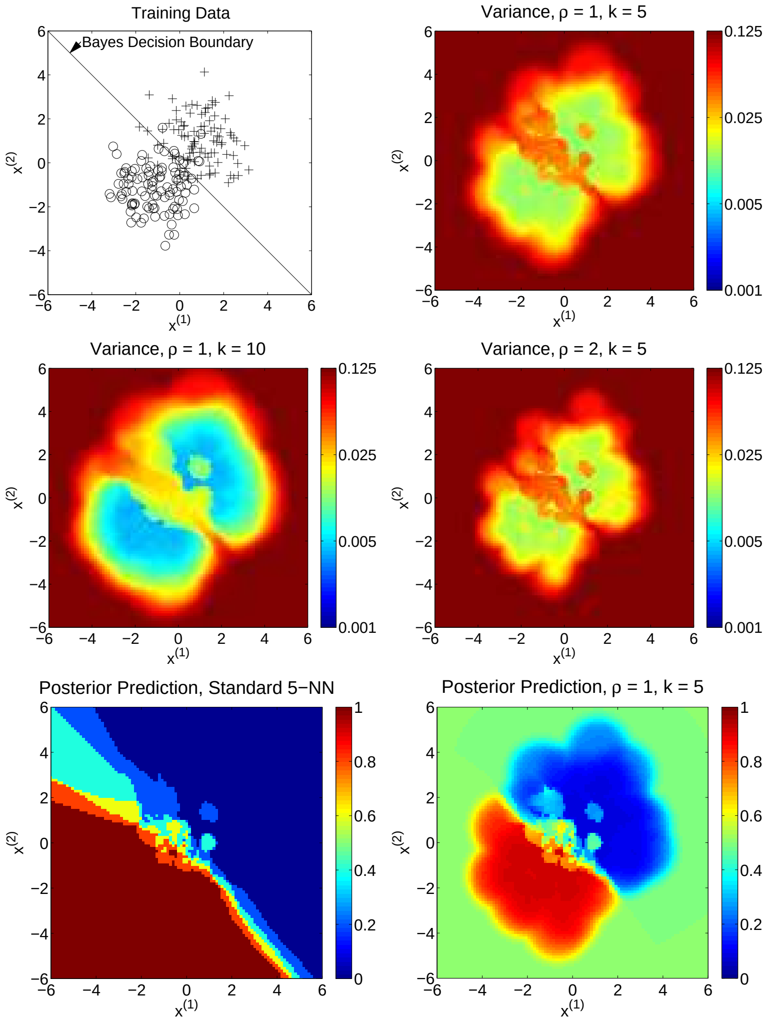


Figure 3.3: [Best viewed in color] A training set $\mathcal{T}_n$ with $n = 200$ is plotted in the upper left panel. The variance of the resulting Beta distributional estimates for probabilistic confidence k-NN with various parameter values of k and ρ is shown, on a logarithmic scale, in the next three panels. In the last row, predictions for the posterior class probabilities of standard k-NN are compared to the point estimates (the mean of the distributional estimates) of probabilistic confidence k-NN. Note that the posterior predictions are very similar in high density regions, in particular near the decision boundary. However, while standard k-NN only offers ambiguity reject, confidence k-NN is also able to detect outliers.

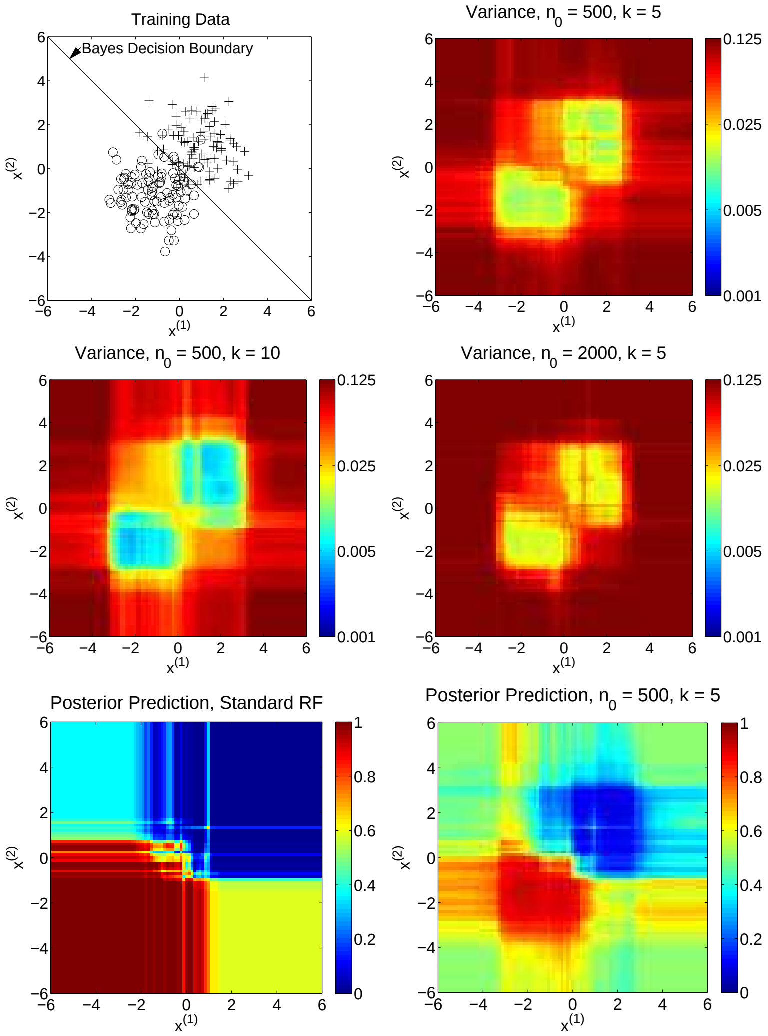


Figure 3.4: [Best viewed in color] The same training set $\mathcal{T}_n$ with $n = 200$ as in Fig. 3.3 is plotted in the upper left panel. The variance of the resulting Beta distributional estimates for CRF with various parameter values of k and n_0 is shown, on a logarithmic scale, in the next three panels. In the last row, predictions for posterior class probabilities of standard RF are compared to the point estimates of CRF.

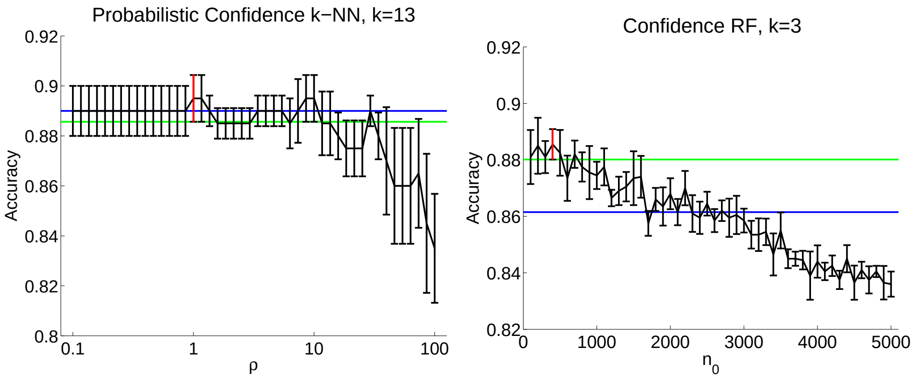


Figure 3.5: Parameter optimization. For fixed k, the classification accuracy of probabilistic confidence k-NN and CRF is estimated by 5-fold cross-validation (CV) for varying ρ and n_0, respectively (black line). In each panel, the black vertical bars indicate the standard error of the estimate, obtained from the CV procedure. The red vertical bar indicates the standard error of the highest accuracy. According to the 1-SE-rule proposed in Section 3.6, we choose the largest regularization parameter (here: largest ρ and n_0) such that the corresponding accuracy is not more than one standard error below the highest estimate for the accuracy (plotted with a green line). Here, this yields $\rho = 10$ and $n_0 = 700$. The blue line is the classification performance of the methods when setting $\rho = 0$ or $n_0 = 0$. Note that, in this example, adding a few hundred points from the auxiliary class 0 slightly increases the accuracy of random forest.

3.8.2 Road Sign Recognition

We now show the usefulness of CRF on a real world problem from road sign recognition. Further, we show that the ordering of the sample points according to the variance of their uncertainty estimates is robust with respect to the exact choice of n_0 and k.

The road signs dataset has been provided by the Robert Bosch GmbH in Hildesheim. It is composed of small gray value images with a resolution of 21×21 pixels. Example images are shown in Fig. 3.6. The training data consists of $n = 3084$ images of the speed limits “50” and “70”, hand labeled as classes 1 and 2, respectively. The classification task is to predict if a test image shows a speed limit of “50” or “70”. However, the interesting feature of this data set is that the test set, of size $n_{test} = 14385$, contains not only yet unseen “50” and “70” speed signs, but also various images not drawn from the same distribution as the training data, in particular other speed signs, other traffic signs, as well as patches of natural images. We show in the following that CRF is able to both correctly classify the 50 and 70 signs, and to automatically detect these various outliers.

For the experiments, we simply represent the images by their gray values, i.e., each image is a vector of dimension $d = 21 \cdot 21 = 441$. In the first experiment, the parameters n_0 and k are chosen according to the procedure presented in Section 3.6, which yields $k = 4$ and $n_0 = 5n = 15420$. $\mathcal{R}$ was chosen such that it covers the

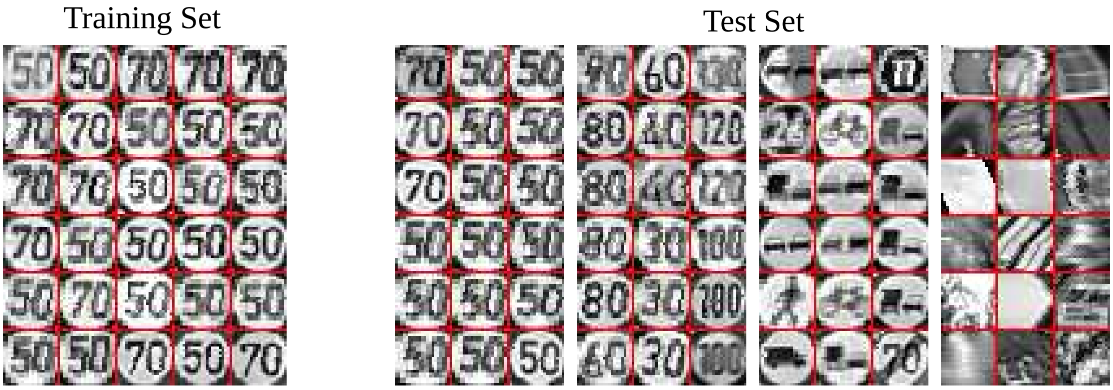


Figure 3.6: Example data of the road sign recognition data set. The left panel shows an excerpt from the training data, consisting of speed limits “50” and “70” only. The remaining panels show the test data (from left to right): speed limit signs “50” and “70” drawn from the same distribution as the training data, speed limit signs other than “50” or “70”, other traffic signs, and finally image patches that do not show a road sign at all.

set $\{x_1, \ldots, x_{3084}\} + [-0.1, 0.1]^{441}$.[5] The classification results on the 14385 test images are shown in Fig. 3.7. First, all images with speed limits “50” (magenta) and “70” (cyan) are classified correctly, i.e., the point estimates for samples of class 1 are lower than 0.5, those for samples of class 2 are higher than 0.5. The variance of the corresponding distributional estimates is very low. The more dissimilar an image is to training data, the higher the variance of its distributional estimate and hence its predicted uncertainty. The variance is relatively low for other speed limits (green), higher for other traffic signs (red) and highest for non-signs (blue). Interestingly, the traffic sign (red) with the lowest variance is an “end of speed limit 70” sign which is quite similar to the speed limit sign itself.

The lower parabola in the figure is the variance of a $Beta(\alpha_1, \alpha_2)$ distribution with parameters α_1, α_2 such that $\alpha_1 + \alpha_2 = k + 1$. It resembles the standard frequentist variance estimate of $\hat{p}(1 - \hat{p})/k$. Note that had we used such a frequentist estimate (disregarding the distances of the query point to the labeled points) rather than our Bayesian approach, then all test points would lie on such a parabola, that is, all points with the same posterior estimate would have the same variance or uncertainty estimate.

In the next experiment, we show the robustness of CRF with respect to the exact choice of the parameters. To this end, we manually vary the two parameters n_0 and k. The results are shown in Fig. 3.8. It can be seen that the correlation between the variances of distributional estimates obtained with different parameter settings is high, i.e., the rank order of the variance of the distributional estimates is quite robust with

[5] As usual, the sum of two sets is defined as $A + B := \{a + b : a \in A, b \in B\}$.

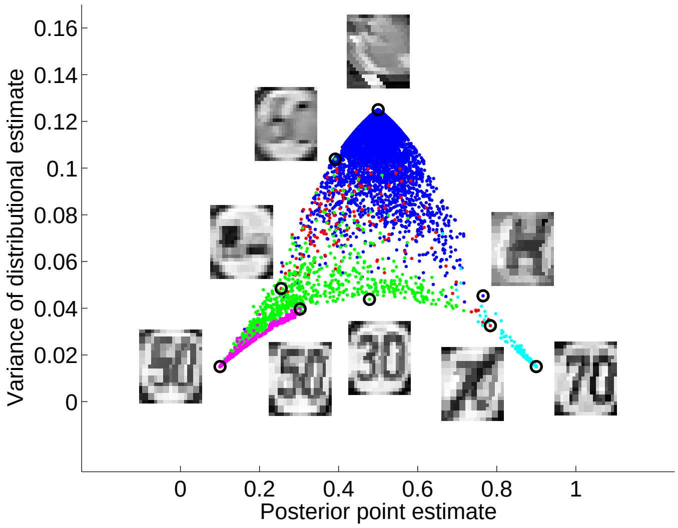


Figure 3.7: Distributional estimates for 14385 test images, represented by the mean and the variance of the estimate. The colors of the dots encode the content of the test image: speed limits “50” and “70” are magenta and cyan, respectively, and *only* these classes were present in the training set (a full-fledged sign recognition system should be trained with *all* traffic signs). Other speed limits are green, other traffic signs are red and non-signs are blue. Some interesting test samples are flagged and the corresponding images are shown nearby. Note that the variance of the prior $Beta(1/2, 1/2)$ is 0.125.

respect to the exact choice of parameter values.

3.8.3 IMS Data

We further illustrate the practical benefit of CRF by another real world application, namely mass spectrometric images [123] of human breast cancer xenografts grown in mice [79]. The data has been provided by Ron M.A. Heeren (FOM-AMOLF, Amsterdam). Imaging mass spectrometry (IMS) is an emerging technology that offers both spatial and spectral resolution and that can simultaneously monitor a large number of (bio-) molecules in organic samples—resulting here in multi-spectral images with more than 4000 channels. Biologically, the data set contains five different regions of interest (necrotic and viable tumor, gelatin, interface region and glass). Individual pixels of the images are labeled based on chemical staining of tissue slices which were cut in parallel to the ones subjected to IMS analysis (see Fig. 3.9). For classifying pixels in new images, rather than working with 4000 dimensions per pixel, only 5 of the most informative spectral channels are chosen for each class, as described in [79]. The resulting data points live in a 12-dimensional space (and not in a 25-dimensional space) because

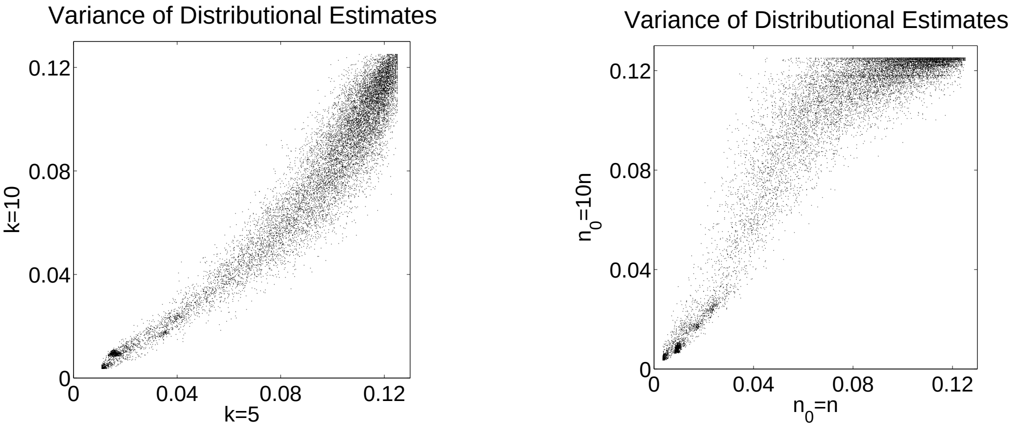


Figure 3.8: Comparison of the variance of the distributional estimates of CRF for different parameter settings. Each dot represents one test sample of the road sign recognition data set. In the left panel, $n_0 = n = 3084$ and k is varied; in the right panel, $k = 10$ and n_0 is varied. Note that the largest attainable variance, namely that of the prior $Beta(1/2, 1/2)$, is 0.125. As already shown e.g. in Fig. 3.4, increasing k and decreasing n_0 lowers the variance of the distributional estimates. The rank order of estimated variances and hence uncertainties is relatively robust with respect to the choice of the parameters k and n_0.

some features are meaningful for more than one class. For an initial analysis, the five tissue (sub-) classes are merged into the two classes "no tumor" (1) and "tumor" (2) as shown in Fig. 3.9. Hence, the task of the classifier is to predict for each pixel in a test image whether the underlying tissue sample is tumorous or not. Five images are used for training, the sixth for testing. The total number of labeled pixels in the training images is $n = 36194$. The parameters k and n_0 are optimized according to the procedure proposed in Section 3.6, the rectangle $\mathcal{R}$ is always chosen such that it covers the set $\{x_1, \ldots, x_{36194}\} + [-0.1, 0.1]^d$.

If standard and confidence RF are trained with the *full* training set, the predictions for $\pi(x)$ are quite accurate for both classifiers (Fig. 3.10). Moreover, the uncertainty of confidence random forests is low in all regions. However, in practice, many real-world training sets are biased or incomplete and it may well happen that a certain tissue type (or e.g. an unknown defect type in the case of industrial optical inspection) is not well represented in the training set. Alternatively, a test sample may accidentally be drawn from a different distribution than the training data. One of the main benefits from our approach is that this can be automatically detected. We hence investigate a scenario in which the test image contains tissue types which are absent from the training data. This is imitated by excluding one of the subclasses from the training set.

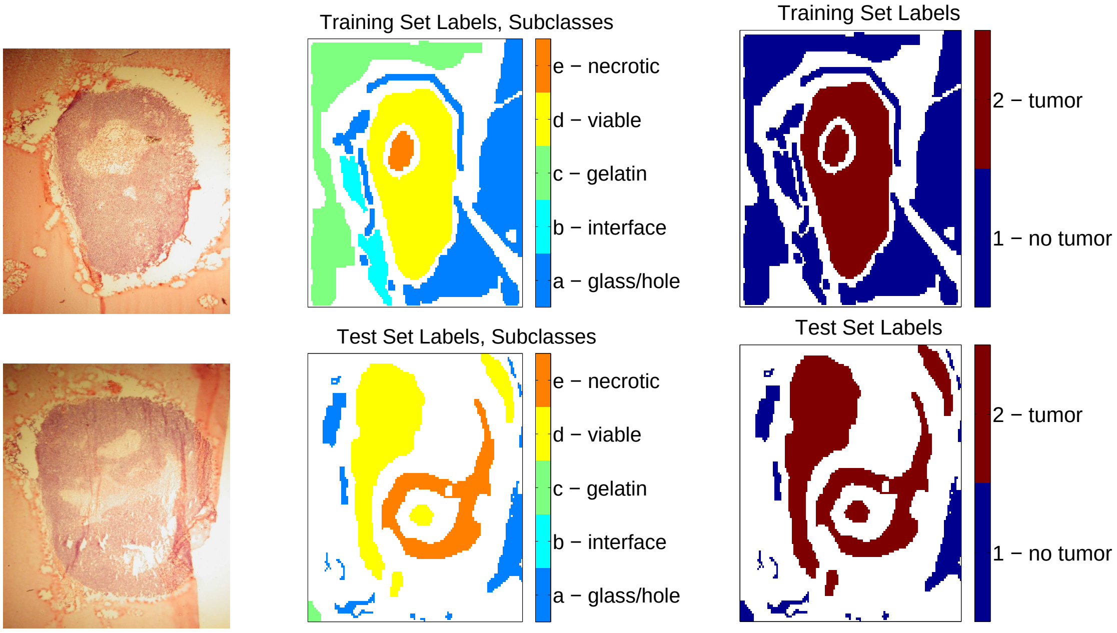


Figure 3.9: [Best viewed in color] The left column shows (parts of) images of chemically stained tissue slices. The expert labels obtained from these are plotted in the middle column, where labels a, b, c, d and e represent “glass/hole”, “interface”, “gelatin”, “viable” and “necrotic”, respectively. The labels in the white regions are missing. In the right column, these labels are merged into “no tumor” (1) and “tumor” (2), the class labels that are actually used for training. The first row shows one of the five training images whereas the second row is the test image.

If subclass “a” is omitted, both standard and confidence RF have very inaccurate posterior predictions at test samples of that subclass. However, the lack of informative training data is detected by confidence RF by a high variance of the second-order distribution at those samples. The exclusion from subclass “c” is also detected by confidence RF. In contrast to omitting “a”, using standard RF would not lead to totally wrong predictions. Posterior class predictions are quite accurate when omitting “e”, because “d” and “e” strongly overlap in feature space. This also explains why the lack of “e” is not detected by confidence RF.

3.9 Conclusions

In this chapter, we have discussed the need for a confidence measure for posterior class estimates and proposed to express this confidence in terms of a second-order distribution.

We derived distributional estimates for ε-nearest neighbors, k-nearest neighbors and

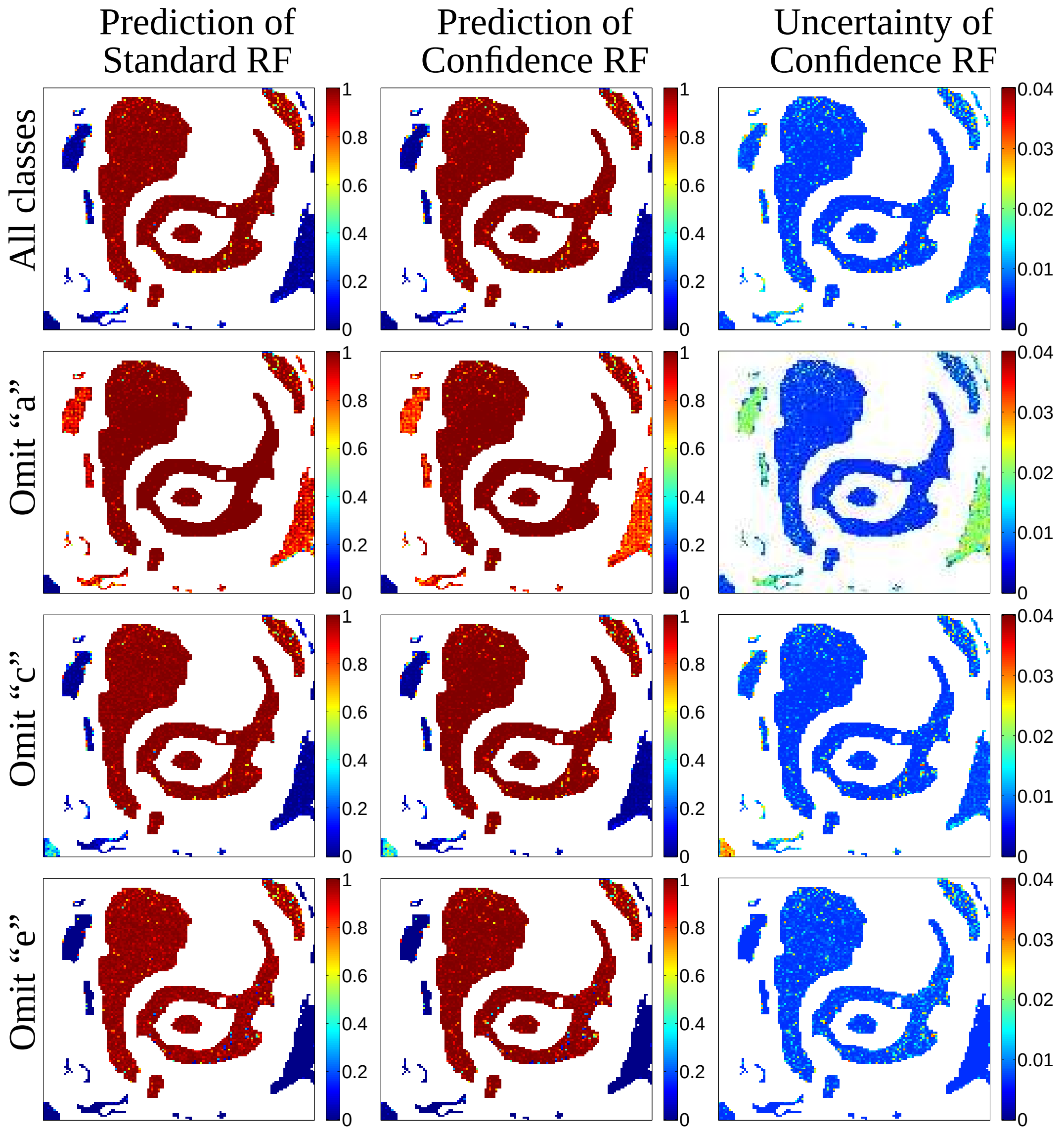


Figure 3.10: [Best viewed in color] Classification results for the labeled regions of the test image shown in Fig. 3.9. Standard RF predictions are shown on the left and the point estimates obtained from confidence random forests in the middle. The last column indicates the uncertainty of this prediction, expressed by the variance of the distributional estimate. The most interesting results are obtained if subclass “a” (glass/hole) is removed. The respective regions in the test image are now classified as tumorous by both classifiers. However, CRF indicates that the respective region in feature space contains only relatively few training points and that the prediction may be erroneous.

random forests and proved the (universal) consistency of these estimates for ε-nearest neighbor and confidence k-NN. It is an open question whether the proposed modification of RF is universally consistent.

Although we concentrated on two-class problems, the derivations can straightforwardly be generalized to multi-class problems, resulting in Dirichlet instead of Beta distributions.

Using two real-world data sets, we demonstrated that the proposed confidence RF algorithm combines the advantages of one- and two-class learning: the distributional estimates indicate to what extent a feature vector is consistent with the training data, allowing to detect outliers or novel subclasses, while the classification accuracy is as high as that of standard RF. The latter is ensured by setting the parameters n_0 and k such that the classification accuracy of CRF is not statistically significantly worse than that of standard RF. Moreover, the detection of outliers was shown to be robust with respect to the exact choice of the parameters n_0 and k of the method.

To the best of our knowledge, this is the first time that auxiliary data is used for *supervised* classification to obtain a confidence measure for posterior predictions. The derived second-order distributions naturally arise from Bayesian inference based on “counting labels” and the models rely on classifiers that are based on this principle. Nevertheless, the idea of using auxiliary data is of course potentially applicable to other classifiers. Highly simplified, the posterior estimate for the auxiliary class can be interpreted as a *heuristic* measure of relative prediction uncertainty.

Interesting open problems are the definition of scoring rules [72] to evaluate distributional estimates, and the proof of consistency of the distributional estimate for confidence RF.

Note that the second-order distributions derived in this chapter are model-based, depending on the parameters of the corresponding method (ε, k, n_0 and/or ρ, respectively), and mainly allow for a relative comparison of prediction uncertainty. In the next chapter, we will derive second-order distributions for kernel density classification that indeed approximate the true sampling distribution

$$F_{n,x}(q) = \mathbf{P}(\hat{\pi}(x) \leq q | \mathcal{T}_n), \quad q \in [0,1] \tag{3.25}$$

and these will be used for active learning. Note the difference between Eqs. (3.1) and (3.25).

3.10 Appendix

Proof of Proposition 3.6. Let $\varepsilon > 0$ and define $\alpha' := \alpha_1/(\alpha_1 + \alpha_2)$. Then,

$$
\begin{aligned}
P(|Z_n - \alpha'| > \varepsilon) &\leq \frac{1}{\varepsilon^2}\mathbf{E}\left(Z_n - \alpha'\right)^2 && (3.26)\\
&= \frac{1}{\varepsilon^2}\mathbf{E}\left(Z_n - \mathbf{E}Z_n + \mathbf{E}Z_n - \alpha'\right)^2\\
&= \frac{1}{\varepsilon^2}\Big(\mathbf{E}(Z_n - \mathbf{E}Z_n)^2 + 2\mathbf{E}(Z_n - \mathbf{E}Z_n)\left(\mathbf{E}Z_n - \alpha'\right) + \left(\mathbf{E}Z_n - \alpha'\right)^2\Big)\\
&= \frac{1}{\varepsilon^2}(V(Z_n) + (\mathbf{E}Z_n - \alpha')^2) && (3.27)
\end{aligned}
$$

where inequality (3.26) follows from Markov's inequality. Plugging in the formulas for the respective moments of a Beta distribution, It readily follows that both summands in (3.27) converge to 0 in probability as $n \to \infty$. This completes the proof. □

Proof of Proposition 3.7. First of all, note that the estimation of $\pi(x)$ can be regarded as a regression problem: If we rename the class labels 1 and 2 as 0 and 1, respectively, then $\pi(x)$ is equal to the regression function $\mathbf{E}[Y|X = x]$. Thus, estimating $\pi(x)$ simply by $\breve{\pi}(x) = n_{2,x}/(n_{1,x} + n_{2,x})$ corresponds to kernel regression with a box kernel. The strong L_1-consistency of kernel regression under the assumptions (3.17) (and taking into account that $Y \in \{0, 1\}$ and that we are using a box kernel) is proven in [49]. Hence, by Remarks 3.3 and 3.4, the estimate $\breve{\pi}(x)$ converges in probability to $\pi(x)$ for μ-almost all x. A necessary condition for this convergence is that $n_{1,x} + n_{2,x} \to \infty$ in probability. Thus, setting $\alpha_{i,n} = n_{i,x}$ in Proposition 3.6, it follows that $Beta(1/2 + n_{2,x}, 1/2 + n_{1,x})$ converges in probability to $\pi(x)$. Hence, the distributional estimate in Eq. (3.3) for ε-nearest neighbor is consistent. □

Proof of Proposition 3.8. As in the proof of Proposition 3.7, note that the estimation of $p_{i|x}$ can be regarded as a regression problem if we rename the class labels 0, 1 and 2. To this end, let $\psi_i(j) = \delta_{ij}$, where δ_{ij} is Kronecker's delta. Further, let Y' (taking values in $\{0, 1, 2\}$) be the label random variable of the extended classification problem including the samples from class 0. Then, $p_{i|x}$ is equal to the regression function $\mathbf{E}[\psi_i(Y')|X = x]$. Hence, estimating $p_{i|x}$ simply by $\breve{p}_{i|x} = n_{i,x}/(n_{0,x} + n_{1,x} + n_{2,x})$ corresponds to standard k-NN regression. The strong L_1-consistency of k-NN regression with bounded response under the assumptions (3.18) is proven in [47]. Hence, by Remarks 3.3 and 3.4, the estimate $\breve{p}_{i|x}$ converges in probability to $p_{i|x}$ for μ-almost all x. It follows that $\breve{\pi}(x) = \breve{p}_{2|x}/(\breve{p}_{1|x} + \breve{p}_{2|x}) = n_{2,x}/(n_{1,x} + n_{2,x})$ converges to $p_{2|x}/(p_{1|x} + p_{2|x}) = \pi(x)$ in probability for μ-almost all x because $f(x_1, x_2) := x_1/(x_1 + x_2)$ is a continuous function for $x_1 + x_2 > 0$. Condition (3.19) ensures that $p_{1|x} + p_{2|x} > 0$ if $p(x) > 0$. A necessary condition for this convergence is that $n_{1,x} + n_{2,x} \to \infty$ in probability. Thus, as in the proof of Proposition 3.7, it follows from Proposition 3.6 that $Beta(1/2+n_{2,x}, 1/2+n_{1,x})$ converges in probability to $\pi(x)$ for μ-almost all x. Hence, the distributional estimate in Eq. (3.8) for k-nearest neighbor is consistent. □

Proof of Proposition 3.9. Again, the estimation of $\pi(x)$ is regarded as a regression problem: If we rename the class labels 1 and 2 as 0 and 1, respectively, then $\pi(x)$ is equal to the regression function $\mathbf{E}[Y|X = x]$. The conditions (3.20) are sufficient for universal consistency of standard nearest neighbor regression for $\rho = 0$ [47]. In *probabilistic* confidence k-NN, the number of neighbors considered from the two original classes is always k. The neighbors are simply weighted by their probability of still being in $B_k^0(x)$ after hypothetically augmenting the training set by samples from a reference distribution. Hence, for the consistency of the method, we only need to ensure that the sum of the weights goes to infinity if k goes to infinity. A sufficient condition is that ρ converges to infinity sufficiently slow so that the individual weights of the neighbors do not tend to 0. First of all, we have that

$$\mathbf{P}(x_{(j)} \in B_k^0(x)) = e^{-\rho V(r_j)} \sum_{i=0}^{k-j} \frac{(\rho V(r_j))^i}{i!} \geq e^{-\rho V(r_j)} \quad \text{for all } j \leq k \tag{3.28}$$

For fixed $\rho \geq 0$, it follows from $\mathbf{E}[|X|] < \infty$ that $\mathbf{E}[\rho V(r_j)] \leq C_1 < \infty$. If ρ increases with n, condition (3.21) ensures that $\mathbf{E}[\rho V(r_j)] \leq C_2 < \infty$, and thus that $\mathbf{E}[e^{-\rho V(r_j)}] \geq \delta > 0$ for some $\delta > 0$. It follows from Eq. (3.28) that $\mathbf{P}(x_{(j)} \in B_k^0(x)) \geq \delta > 0$. □

Proof of Proposition 3.10. The proof is similar to that of Proposition 3.8. Given the assumptions (3.22) and (3.24), the strong L_1-consistency of $\breve{\pi}(x) = n_{i,x}/(n_{0,x} + n_{1,x} + n_{2,x}), i = 0, 1, 2,$ is shown in [116].[6] Hence, by Remarks 3.4 and 3.3, the estimate $\breve{p}_{i|x}$ converges in probability to $p_{i|x}$ for μ-almost all x. It follows that $\breve{\pi}(x) = \breve{p}_{2|x}/(\breve{p}_{1|x} + \breve{p}_{2|x}) = n_{2,x}/(n_{1,x} + n_{2,x})$ converges to $p_{2|x}/(p_{1|x} + p_{2|x}) = \pi(x)$ in probability for μ-almost all x because $f(x_1, x_2) := x_1/(x_1+x_2)$ is a continuous function for $x_1+x_2 > 0$. Condition (3.23) ensures that $p_{1|x}+p_{2|x} > 0$ if $p(x) > 0$. A necessary condition for this convergence is that $n_{1,x} + n_{2,x} \to \infty$ in probability. As in the proofs of Proposition 3.7 and 3.8, it follows from Proposition 3.6 that $Beta(1/2 + n_{2,x}, 1/2 + n_{1,x})$ converges in probability to $\pi(x)$ (set $\alpha_{i,n} = n_{i,x}/n$ in Proposition 3.6) for μ-almost all x. Hence, the distributional estimate in Eq. (3.14) for a single tree is consistent. □

Proposition 3.11. *Let $Q = (Q_0, Q_1, Q_2) \sim Dir(\alpha_0, \alpha_1, \alpha_2)$ be a Dirichlet distributed random vector with parameters α_0, α_1 and α_2. Then*

$$\frac{Q_2}{Q_1 + Q_2} \sim B(\alpha_2, \alpha_1)$$

[6] According to Theorem 3 in [116], the conditions of Proposition 3.10 imply those of Theorem 2 in [116]. The strong L_1-consistency is an intermediate result in the proof of Theorem 2 in [116].

where $B(\alpha_2, \alpha_1)$ is the Beta distribution with parameters α_2 and α_1.

Proof. Let $Y_i, i = 0, 1, 2$, be stochastically independent and Gamma distributed with shape parameter α_i and scale parameter 1, i.e. $Y_i \sim \Gamma(\alpha_i, 1), i = 0, 1, 2$. It is shown in [130] that

$$\left(\frac{Y_0}{Y_0 + Y_1 + Y_2}, \frac{Y_1}{Y_0 + Y_1 + Y_2}, \frac{Y_2}{Y_0 + Y_1 + Y_2}\right) \sim Dir(\alpha_0, \alpha_1, \alpha_2) \tag{3.29}$$

If we define

$$Q_i := \frac{Y_i}{Y_0 + Y_1 + Y_2}, i = 0, 1, 2$$

it follows that

$$(Q_0, Q_1, Q_2) \sim Dir(\alpha_0, \alpha_1, \alpha_2)$$

and

$$\frac{Q_2}{Q_1 + Q_2} = \frac{\frac{Y_2}{Y_0+Y_1+Y_2}}{\frac{Y_1+Y_2}{Y_0+Y_1+Y_2}} = \frac{Y_2}{Y_1 + Y_2} \overset{(*)}{\sim} B(\alpha_2, \alpha_1)$$

where statement $(*)$ is a special case of the statement in (3.29). □

4 Distributional Estimate Active Learning

Active learning techniques aim at reducing the labeling effort for classifier training by querying labels for those samples which are most important for achieving a sufficient classification performance. Informative samples can be close to the decision boundary or in unexplored regions of feature space. Additionally, the density of the underlying class distributions at a training sample is highly relevant for classifier performance. In this chapter, we propose a novel active learning strategy that trades off these three criteria in a principled way by using a second-order distributional estimate for the posterior class probabilities at unlabeled points. The mean of such a distribution corresponds to the usual point estimate, whereas the spread of the distribution measures the confidence in this estimate and thus encodes the degree of exploration in that region of feature space. A comprehensive comparison using real-world data sets from UCI [9], Caltech-4 [60] and USPS Zip Corpus [102] shows the superiority of the proposed AL strategy compared to random sampling, uncertainty sampling and an approach previously proposed by Lindenbaum et al. [111].

4.1 Introduction

In the common setting of supervised learning, a set of *labeled* samples is required for training a classifier. However, the labeling process itself often is difficult, expensive or time-consuming as human expertise usually is indispensable. Hence, if *unlabeled* samples are available in great quantities (consider applications such as speech recognition, defect detection or web page classification), only a small subset of this data can be annotated. If this subset is chosen randomly (referred to as "passive learning" or "random sampling" in the following), a lot of effort may be wasted for labeling those samples which do not contribute much to the final classification performance. To obtain low prediction errors with few labels, the subset selection may be guided by *active learning* (AL) approaches. AL is an iterative process that sequentially chooses the samples to be labeled using information extracted from previously labeled samples and possibly from the (large) pool of unlabeled data. At the end of the AL process, the final classifier is usually trained on the labeled data in a supervised fashion, discarding the unlabeled set.

There are at least three important criteria when evaluating the utility of the label for a yet unlabeled instance:

(i) the distance to the current decision boundary,

(ii) the density of the marginal distribution of features at that point,

(iii) the number of labeled instances in the neighborhood.

In the above, the notion of "distance" and "neighborhood" depends on the employed classifier and metric. Ceteris paribus, each piece of information contributes to an appropriate ranking of the unlabeled samples: (i) the higher the uncertainty in the current label prediction, (ii) the higher the density, and (iii) the less explored a region in feature space, the more interesting it is to acquire a new label in that region. Note that the relative emphasis on the criteria (i) and (iii) governs the trade-off between "exploitation" and "exploration", namely labeling instances near the decision boundary in order to refine it versus labeling instances in regions of feature space that contain no or a few labeled samples only.

The key properties of the AL strategy presented in this chapter are taking into account all three criteria in a unified statistically principled way and having only linear time complexity per iteration step in the number of unlabeled examples. The prerequisite is a classifier that outputs not only a point estimate for the posterior class probabilities at each point in feature space, but a second-order distributional estimate over the posterior class probabilities (see Chapter 3). The mean of this distributional estimate corresponds to the usual point estimate for the posterior (and thus is a measure for the distance to the current decision boundary), whereas the spread of the distribution reflects the confidence in the estimated posterior and thus considers the number of labeled instances in the neighborhood. As will be explained in detail in Section 4.3, other approaches either do not consider all three criteria, have higher computational complexity or consider the trade-off between exploitation and exploration by data preprocessing or by additional model parameters. To the best of our knowledge, our suggested AL strategy is the first to make use of distributional estimates. We thus naturally refer to it as "DEAL" (Distributional Estimate Active Learning).

In Section 4.2, we define the exact AL setup considered. In Section 4.3, we review some existing AL methods and thereby motivate the proposed approach, which is presented in detail in Section 4.4. The implementation of the strategy using kernel density classification is presented in Section 4.5, followed by the corresponding results in Section 4.6.

4.2 Problem Setup

We consider the following common classification scenario. Assume a random vector (X, Y) that has the joint probability density $p(x, y)$, where $x \in \mathcal{X}$ is a feature vector and $y \in \mathcal{Y}$ its class label. In this chapter, for simplicity, we focus on a binary classification setting with $\mathcal{Y} = \{1, 2\}$ and $\mathcal{X} \subseteq \mathbb{R}^d$.

Let $L_{ij}, i, j = 1, 2$, denote the loss of classifying a point of class i as j. The expected loss of a classifier $h : \mathcal{X} \to \mathcal{Y}$ is defined as

$$\begin{aligned}\mathbf{E}_{(X,Y)}[Loss(X,Y)|h] &= \int_{R_1(h)} L_{21} p(x, Y = 2) dx + \int_{R_2(h)} L_{12} p(x, Y = 1) dx \quad (4.1)\\ &= \int_{R_1(h)} L_{21} p(Y = 2|x) p(x) dx + \int_{R_2(h)} L_{12} p(Y = 1|x) p(x) dx\end{aligned}$$

where $R_1(h)$ and $R_2(h)$ are those regions in feature space where h assigns class label 1 or 2, respectively. It can be easily verified that the Bayes classifier

$$h_B(x) = \arg\max_{y=1,2} p(x, y) L_{y,3-y} \quad (4.2)$$

minimizes the expected loss. It assigns class 2 iff $p(Y = 2|x) > \theta$, where $\theta = L_{12}/(L_{12} + L_{21})$. In practical situations, the posterior $p(Y = 2|x)$ is, of course, not known and $h_B(x)$ needs to be estimated from training instances.

The setting considered here is *pool-based* AL, where we start with a small (possibly empty) set $\mathcal{L} = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ of labeled data and a large pool $\mathcal{U} = \{x_{l+1}, \ldots, x_{l+u}\}$ of unlabeled data. The main assumption is that the feature vectors $x_1, \ldots, x_{l+u}$ are i.i.d. realizations from the marginal density $p(x)$. The labeled realizations from (X, Y) in $\mathcal{L}$, in contrast, do not need to be independent since the labeled samples selected later on by the AL strategy are not independent, either.

Throughout this chapter, we assume that only one label is queried at a time[1] and that annotation costs are equal for all instances.

Remark 4.1. *Note that, even if the elements of $\mathcal{L}$ are stochastically independent prior to the AL process, in general, they become dependent through AL. Hence, the estimates for the posterior class probabilities $p(Y = 2|x)$ may be quite inaccurate. However, it is not necessary to know $p(Y = 2|x)$ exactly in order to coincide with the Bayes classifier $h_B(x)$. Instead, it is sufficient to know whether $p(Y = 2|x)$ is larger than θ or not. Hence, most of the labels for samples in regions where the posterior is clearly*

[1] Another variant considered in Chapter 6 in the context of industrial quality control is batch mode AL [75], [84], where several instances are chosen at the same time at each iteration; this speeds up the AL process at the cost of lower AL performance due to possible overlapping information of the labels queried at the same time.

above or below θ are not important for training a classifier with good performance. This fundamental insight is the basic working principle of AL.

4.3 Related Work

In this section, we review some pool-based AL approaches and thereby motivate the DEAL approach. Most AL strategies can be assigned to one of two groups, based on their definition of the TUV:

1. Approaches without look-ahead step:

   Train a classifier on $\mathcal{L}$ and evaluate it at all $x \in \mathcal{U}$. The TUV of x depends on the respective estimate for the posterior class probabilities $p(y|x)$. Corresponding methods are for example proposed in [1], [40], [55], [62], [86], [89], [104], [105], [106], [122], [125], [131], [157], [164], [165], [176] or [190].

2. Approaches with one-step look-ahead:

   Train a classifier on $\mathcal{L}$ and evaluate it at all $x \in \mathcal{U}$. For each instance $x \in \mathcal{U}$ and for each possible class label y, add (x, y) to the current training set $\mathcal{L}$, train the classifier anew and evaluate it at all $x' \in \mathcal{U}\backslash\{x\}$. The TUV of x depends on (the difference between the old and) the new classification output at the instances in $\mathcal{U}\backslash\{x\}$ when $(x, i_1), \dots, (x, i_{|\mathcal{Y}|}), i_j \in \mathcal{Y}$, is added to the training set (usually weighted by the estimated posterior $p(i_j|x)$). Corresponding methods are for example proposed in [73], [111][2] or [153].

4.3.1 Approaches Without Look-Ahead Step

The simplest and probably most commonly used strategy without look-ahead is “uncertainty sampling”. The underlying idea is to query the label for that instance in $\mathcal{U}$ whose current prediction for the posterior class probabilities is closest to θ [105]. A variant of uncertainty sampling for support vector machines is to request the label for that instance which is closest to the decision boundary [176]. Uncertainty sampling methods can be modified by weighting the TUV of x with the density $p(x)$ [164], [188], which is estimated from both labeled and unlabeled instances.[3] Another variant without look-ahead

[2] In [111], the number of look-ahead steps is a user-defined parameter k, but the simulations—even for $k = 2$—are so time-consuming that the strategy is evaluated mainly for $k = 1$ and on very small data sets for $k = 2$.

[3] The cost of a naive implementation of density estimation is of order $O((l + u)^2)$, but the density only needs to be estimated once prior to the AL process. The estimates can be stored and thus the density lookup does not affect the considerations below about the computational complexity for a single query.

step is “query-by-committee” [62], [165], where a label is requested at the instance where a committee of classifiers disagrees most. This principle is also applied in [1] utilizing bagging and boosting techniques.

Strategy	Criterion (i)	(ii)	(iii)	Computational complexity
Random Sampling	-	(✓)	-	-
Uncertainty Sampling [40], [55], [89], [104], [105], [157], [176]	✓	-	-	low
Density-Based Approaches [164], [188]	✓	✓	-	low
Query-by-Committee [1], [62], [122], [125], [131], [165], [190]	✓	-	-	low
One-step look-ahead [73], [111], [153]	(✓)	(✓)	(✓)	high
DEAL	✓	✓	✓	low

Table 4.1: Comparison of different kinds of AL sampling strategies with respect to considering three important criteria when querying a label for an instance: (i) distance from the current decision boundary, (ii) density of the marginal distribution of features, and (iii) the number of labeled points in the neighborhood. The low complexity methods are those without look-ahead step. Further explanations and the exact computational complexities are given in Section 4.3.3. Note that random sampling implicitly selects more samples in regions with higher density.

As summarized in Table 4.1, none of these algorithms without look-ahead step incorporates all three query criteria stated in the introduction. First, uncertainty sampling methods only take into account the distance to the decision boundary, which is measured by inducing a pseudometric on feature space: the distance between two points is quantified by the absolute difference of their posterior class probabilities. Second, density-based approaches obviously incorporate density information but still do not take into account the number of labeled points in the neighborhood of x. Finally, query-by-committee algorithms consider the distance to the decision boundary implicitly: Different members of a committee disagree more on the prediction at an instance x if it is close to the decision boundary. However, more and more labels for instances in regions with a well-established decision boundary may be requested and exploration of feature space is neglected. Moreover, density information is not considered in the standard setting of query-by-committee.

Several proposals have been made to avoid that too greedy an AL strategy overlooks large regions in feature space. In [138], the AL strategy switches between an exploration and an exploitation step with a certain probability, whereas in [31], the TUV is a weighted mean of an exploration and an exploitation term. Other approaches cluster the data once prior to the AL process [64], several times during AL [134] or before each iteration step [15]. All this can help to avoid to completely overlook large regions in feature space that would be classified incorrectly by too greedy a refining of the decision boundary.

4.3.2 Approaches With One-Step Look-Ahead

An approach with look-ahead step is proposed in [153], where the TUV of x is large iff the posterior predictions at the points in $\mathcal{U}\backslash\{x\}$ are far from 0.5 when x is added to the training set. This idea is combined with SSL in [193]. In [73], the TUV of x is defined as the resulting expected change in the classifier output at the points in $\mathcal{U}\backslash\{x\}$ when x is added to the training set. AL strategies for 1-nearest neighbor classification are presented in [111], where the two proposed definitions of the TUV are similar to the ones in [153] and [73].

The above look-ahead methods optimize the choice on the instance to be labeled next by brute force and thereby implicitly consider the three query criteria.

4.3.3 Comparing the Groups

The computational cost of the AL strategies with look-ahead step are substantially higher due to the additional loop. To be more precise, let $f_{train}(l)$ and $f_{test}(l)$ be the computational complexity for training a particular classifier with l samples and evaluating it at one test point[4], respectively. Then, the computational complexity for choosing a point to be labeled is $O(f_{train}(l) + f_{test}(l)u)$ for methods without and $O(f_{train}(l) + f_{test}(l)u + u|\mathcal{Y}|(f_{train}(l+1) + (u-1) \cdot f_{test}(l+1))) = O(u|\mathcal{Y}|f_{train}(l+1) + u^2|\mathcal{Y}|f_{test}(l+1))$ for methods with look-ahead step.[5] Simply put, methods without look-ahead have *linear* complexity in the number of unlabeled points u, whereas the complexity of methods with look-ahead is *quadratic* in u.

The slower look-ahead methods empirically seem to perform better than the approaches without look-ahead step. The authors of [153] show this for their strategy by comparing it to uncertainty sampling and query-by-committee, where all methods are implemented based on a naïve Bayes classifier. In [111], the proposed look-ahead approach is compared to two different variants of uncertainty sampling and performs best on three out of the four investigated real-world data sets.

[4] Note that the evaluation of a classifier may depend on the number of training points. Hence, f_{test} may depend on l.

[5] Note that $O(f(l)) \neq O(f(l+1))$ if f increases exponentially with l.

4.3.4 Comparison to DEAL

We propose an AL strategy that, on one hand, has the computational complexity of methods without look-ahead step, and, on the other hand, considers all three selection criteria stated above in a principled way. The latter is achieved in a unified framework without the necessity of clustering the data or switching explicitly between exploitation and exploration. Instead, we employ a single classifier that provides not only an estimate for the posterior class probabilities but also an estimate for the *uncertainty* of that estimate which depends on the number of labeled instances in the neighborhood. As assumed in the next section, we require a *distributional* estimate for the posterior class probability $p(Y = 2|x)$, i.e. a second-order distribution whose mean corresponds to the posterior point estimate and thus is related to the distance of x to the decision boundary and whose spread is related to the number of labeled points in the neighborhood of x. An example of how such a distribution can be obtained is presented in Section 4.5.

4.4 Active Learning with Distributional Estimates

4.4.1 Assumptions

As defined in Section 4.2, let $\mathcal{L} = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ be a set of labeled samples and $\mathcal{U} = \{x_{l+1}, \ldots, x_{l+u}\}$ a large set of unlabeled observations. For each labeled set $\mathcal{L}$, we assume the existence of a classifier that for any $x \in \mathcal{X}$ provides us not only with a point estimate for the posterior class probabilities, but in fact with a *distributional* estimate of the posterior class probabilities. To simplify notation, we tacitly assume that this estimate has a density

$$g_{2|x}(q) = \frac{d}{dq}\mathbf{P}(\hat{p}(Y = 2|x) \leq q), \quad q \in [0, 1]$$

and refer to $g_{2|x}$ as distributional estimate. Examples for $g_{2|x}$ are shown in Fig. 4.1. However, note that the AL strategy is defined for *any* distribution and we will use an example of a discrete distribution below to motivate a definition.

The distributional estimate $g_{2|x}$ encodes our *uncertainty* in the predicted class probability at x, in particular for each $x \in \mathcal{U}$, and plays a key role in our AL strategy. In addition, as $l + u \gg 1$, we assume that the marginal $p(x)$ can be estimated, e.g. via some non-parametric density estimate $\hat{p}(x)$.

4.4.2 Classifier Losses and Their Estimates

The expected loss of a classifier h is defined in Eq. (4.1). The *local* loss at any $x \in \mathcal{U}$ is given by

$$R_x(h) = p(x)\left[L_{21}p(Y=2|x)\mathbf{1}\{h(x)=1\} + L_{12}p(Y=1|x)\mathbf{1}\{h(x)=2\}\right]$$

where the indicator function $\mathbf{1}\{s\}$ equals 1 if s is true and 0 otherwise. In this subsection, we define two different estimates of the local loss, one based on point estimates of posterior class probabilities and one based on distributional estimates. These two definitions are crucial ingredients for the definition of the TUV in the DEAL strategy.

First of all, let

$$\hat{p}(Y=2|x) = \int_0^1 q\, g_{2|x}(q)dq \tag{4.3}$$

be the point estimate for the posterior class probability corresponding to the distribution $g_{2|x}$. Given this point estimate, the classifier that minimizes the expected loss (globally and locally) is

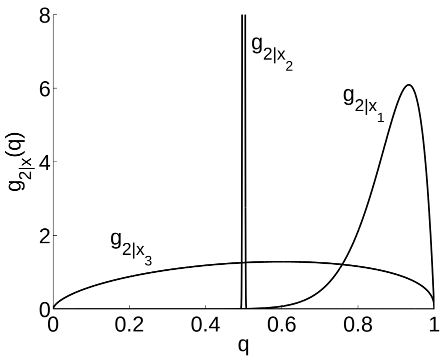
$$h_o(x|g_{2|x}) = \begin{cases} 2 & \text{if } \hat{p}(Y=2|x) > \theta \\ 1 & \text{otherwise} \end{cases} \tag{4.4}$$

and the usual plug-in estimate of the expected local loss of a point estimate is given by

$$\hat{R}_x(\hat{p}(Y=2|x)) = \hat{p}(x)\min\{\hat{p}(Y=2|x)L_{21}, \hat{p}(Y=1|x)L_{12}\} \tag{4.5}$$

To motivate the definition of the expected local loss of a distributional estimate, consider the following artificial discrete example with $\mathbf{P}(\hat{p}(Y=2|x)=0.2) = 0.5 = \mathbf{P}(\hat{p}(Y=2|x)=0.9)$, $\theta = 0.5$ and $p(x) = 1$. It follows from Eq. (4.3) that $\hat{p}(Y=2|x) = 0.55$; we assign class 2 (according to Eq. (4.4)) and expect to incur a local loss of 0.45 at x (Eq. (4.5)). This is indeed the best we can achieve if we have to take a classification decision based on the current information. But, given $g_{2|x}$, what local loss do we expect to incur on average if we had more information, e.g. if we had the opportunity to query some labels at x or in its neighborhood? More specifically, what local loss do we expect to incur on average, given the current information based on the distributional estimate, if we knew the true posterior instead of having a distributional estimate of it? Or, from a different point of view, what local loss do we expect to incur if we had the possibility to label so many points in the neighborhood of x that we can be almost sure to predict the same class as the Bayes classifier (Eq. (4.2))? In this simplified example, where the posterior is either 0.2 or 0.9 with equal probability, the expected local loss would then be $0.5 \cdot 0.2 + 0.5 \cdot 0.1 = 0.15$.

Transferring these considerations to a distributional estimate with density $g_{2|x}$, where



DEAL	$R_x(\hat{p}(Y=2\|x))$	$R_x(g_{2\|x})$	$TUV(x)$
x_1	0.118	0.118	$1.7 \cdot 10^{-5}$
x_2	0.5	0.499	0.001
x_3	0.467	0.286	0.181

Uncertainty Sampling	$TUV(x)$
x_1	0.118
x_2	0.5
x_3	0.467

Figure 4.1: Examples of distributional estimates $g_{2|x}$ at three unlabeled points x_1, x_2, x_3. The TUV of the DEAL strategy is large for $g_{2|x_3}$ and small for both $g_{2|x_1}$ and $g_{2|x_2}$. In contrast, the TUV of uncertainty sampling (expressed according to Eq. (2.2)) is larger for $g_{2|x_2}$ than for $g_{2|x_3}$, as $\hat{p}(Y=2|x_2) = 0.5$ and $\hat{p}(Y=2|x_3) = 0.533$.

summation becomes integration, yields

$$\hat{R}_x(g_{2|x}) = \hat{p}(x) \int_0^1 g_{2|x}(q)[qL_{21}\mathbf{1}\{q \le \theta\} + (1-q)L_{12}\mathbf{1}\{q > \theta\}]dq \qquad (4.6)$$

$$= \hat{p}(x) \int_0^1 g_{2|x}(q)\hat{R}_x(q)dq \qquad (4.7)$$

for the expected local loss of a distributional estimate $g_{2|x}$.

4.4.3 Proposed Active Learning Strategy

In the previous subsection, we defined the expected local loss of a point estimate $\hat{p}(Y = 2|x)$ (Eq. (4.5)) and that of a distributional estimate $g_{2|x}(q)$ (Eq. (4.6)). Next, we propose an AL criterion based on these two quantities. To motivate the approach, consider the distributional estimates for three different unlabeled instances x_1, x_2 and x_3 shown in Fig. 4.1. Assuming that $\theta = 0.5$ and equal density $\hat{p}(x)$ at all three points, which of these instances should be labeled next?

It is not very interesting to query a label at x_1. First, $\hat{p}(Y = 2|x_1)$ is relatively close to 1, i.e., x_1 is relatively far from the decision boundary. Second and more important, the probability that the true posterior $p(Y = 2|x)$ is smaller than 0.5 is almost 0 according to the distributional estimate $g_{2|x_1}$. Hence, according to Remark 4.1, it is not sensible to query a label at x_1 because all we need to know about $p(Y = 2|x_1)$ is if it is larger or smaller than 0.5. Requesting a label at x_2 is not very sensible either, although this point is very close to the decision boundary: Even if the additional label at x_2 influenced the final class prediction in the neighborhood of x_2, this would have little impact on the final classification performance. This is because we can be very certain that the true posterior $p(Y = 2|x_2)$ is very close to 0.5 according to the distributional estimate $g_{2|x_2}$. At x_3, based on the (current) point estimate $\hat{p}(Y = 2|x_3) = 0.53$, we assign class 2. But as there is a relatively high probability that $p(Y = 2|x_3)$ is small, which would lead to a large local loss at x_3, an additional label may have a large impact on the local loss. In particular, it is more advisable to query a label at x_3 than at x_2 although the posterior estimate at x_3 is further from 0.5 than at x_2.

The expected local loss of the point and the distributional estimate at x_1, x_2 and x_3 are presented in the table on the right hand side of Fig. 4.1. Based on these examples, we observe that the knowledge about the true posterior class probability $p(Y = 2|x)$ is close to optimal if the difference $R_x(\hat{p}(Y = 2|x)) - \hat{R}_x(g_{2|x})$ is small. The larger $R_x(\hat{p}(Y = 2|x)) - \hat{R}_x(g_{2|x})$, the larger is the expected decrease of the local loss at the point x when knowing the posterior exactly instead of $g_{2|x}$ only, and hence the more interesting an additional label at x becomes. This observation motivates the following definition of the TUV, which is the central expression of this chapter:

$$TUV(x) := \hat{R}_x(\hat{p}(Y = 2|x)) - \hat{R}_x(g_{2|x}) \tag{4.8}$$

We prove in Proposition 4.2 in the appendix that the TUV defined in Eq. (4.8) has the following properties:

$$TUV(x) \geq 0 \tag{4.9}$$

and

$$TUV(x) = 0 \qquad \Leftrightarrow \qquad \int_0^\theta g_{2|x}(q)dq = 1 \text{ or } \int_\theta^1 g_{2|x}(q)dq = 1 \tag{4.10}$$

Eq. (4.9) means that the information gain from a label is always non-negative. Further, Eq. (4.10) exactly reflects Remark 4.1: For an optimal class prediction, exact knowledge of the true posterior is not needed. It suffices to know only whether it is larger or smaller than θ. The fact that the whole mass of $g_{2|x}$ is concentrated either below *or* above θ indicates that indeed sufficiently many labels have been queried in the neighborhood of x and that querying a label at x is a waste of resources.

4.4.4 Beta Distributional Estimates

To make things more explicit, we assume that $g_{2|x}(q) = g_{2|x}(q|a, b)$ follows a Beta distribution with parameters a and b. This distribution family arises as the result of the derivations in Section 4.5. The probability density functions of the Beta distribution for different parameters settings are plotted in Fig. 4.2.

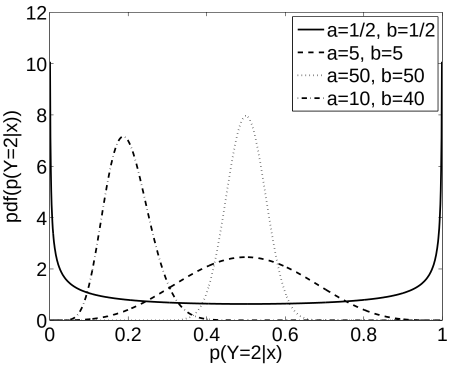


Figure 4.2: Probability density function of the Beta distribution for different parameters a and b. If a and b are approximately equal, the point estimate of the posterior class probability $p(Y = 2|x)$ is close to 0.5. The confidence in this estimate is high if a and b are large, whereas the confidence is low for small a and b. The larger the difference between a and b, the closer the point estimates of $p(Y = 2|x)$ are to 0 or 1. Typically, the sum of a and b will be relatively large (small) if x is in a region of relatively high (low) density with respect to the current set of labeled samples.

The expected value of a Beta distribution equals

$$\hat{p}(Y = 2|x) = \frac{a}{a+b}$$

and the corresponding expected local loss then is

$$\hat{R}_x(\hat{p}(Y = 2|x)|a, b) = \hat{p}(x) \min\left(\frac{aL_{21}}{a+b}, \frac{bL_{12}}{a+b}\right)$$

The expected local loss of the distribution is given by

$$\hat{R}_x(g_{2|x}(q|a,b)) = \hat{p}(x)\left[\int_0^1 \hat{g}_{2|x}(q|a,b)[qL_{21}\mathbf{1}\{q < \theta\} + (1-q)L_{12}\mathbf{1}\{q > \theta\}]dq\right] \tag{4.11}$$

$$= \hat{p}(x)\left[\frac{aL_{21}}{a+b}I_\theta(a+1,b) + \frac{bL_{12}}{a+b}I_{1-\theta}(b+1,a)\right] \tag{4.12}$$

where $I_\theta(a, b)$ is the incomplete Beta function with parameters a and b at position θ (see e.g. [2, chap. 26.5] or Appendix A in Section 4.8). The calculations to obtain term (4.12)

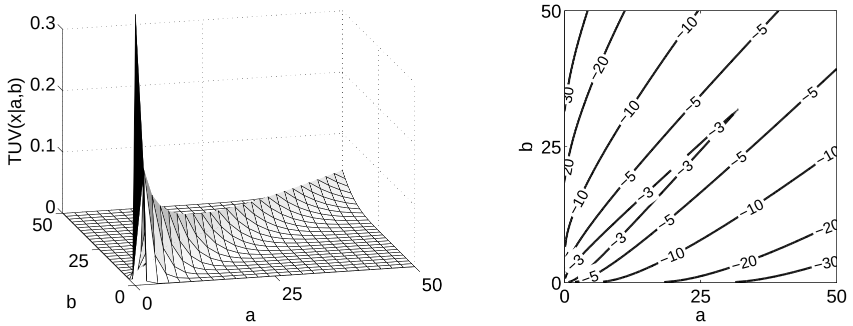


Figure 4.3: The training utility value $TUV(x|a,b) = \hat{R}_x(\hat{p}(Y=2|x)|a,b) - R_x(\hat{g}_{2|x}(q|a,b))$ (Eq. (4.13)), where a and b are the parameters of a Beta distribution. Some examples with different parameter settings are plotted in Fig. 4.2. In both panels, $\hat{p}(x) = 1$ and $\theta = 0.5$; the right panel shows the contour lines of the natural logarithm of the function. Given equal density $\hat{p}(x)$, we prefer requesting a label for a point x at which the point estimate for the posterior is close to 0.5 and the variance of the distributional estimate is high (a approximately equals b and both values are small) over a point x with equal posteriors and low variance (a approximately equals b and both values are large) over a point x where the posterior estimates are far away from 0.5 (a is much larger than b or vice versa).

from term (4.11) are shown in Appendix A in Section 4.8. Finally,

$$\begin{aligned} TUV(x|a,b) &= \hat{R}_x(\hat{p}(Y=2|x)|a,b) - \hat{R}_x(g_{2|x}(q|a,b)) \\ &= \hat{p}(x)\left[\min\left(\frac{aL_{21}}{a+b},\frac{bL_{12}}{a+b}\right) - \frac{aL_{21}}{a+b}I_\theta(a+1,b) - \frac{bL_{12}}{a+b}I_{1-\theta}(b+1,a)\right] \end{aligned} \tag{4.13}$$

The TUV for different parameters a and b is plotted in Fig. 4.3. If a and b are small and approximately equal, then the posterior estimate is close to 0.5 and the variance of the distributional estimate is high; thus the TUV is large. The TUV is small if either a and b are unequal (resulting in point estimates that differ from 0.5) or their sum is large (resulting in a low variance of the distributional estimate). The dependence on the mean of the Beta distribution and the sum of its parameters is made explicit in Fig. 4.4 to compare the TUV of DEAL to that of uncertainty sampling.

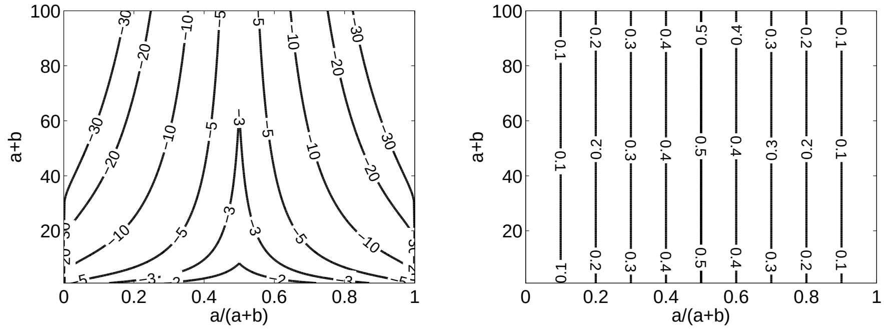


Figure 4.4: The left panel shows the contour lines of the natural logarithm of the training utility value, i.e. the contour lines of the logarithm of $TUV(x|a,b) = \hat{R}_x(\hat{p}(Y = 2|x)|a,b) - \hat{R}_x(\hat{g}_{2|x}(q|a,b))$, in dependence of the mean of the Beta distribution and the sum of its parameters, which are measures of the distance to the decision boundary and the prediction uncertainty, respectively. It is assumed that $\hat{p}(x) = 1$ and $\theta = 0.5$. If $a/(a+b)$ is fixed, the larger $a+b$, the less important the label at the respective point in feature space. If $a+b$ is fixed, the closer $a/(a+b)$ to 0.5, the more important a label. The contour lines for the TUV of uncertainty sampling are shown in the right panel. We have $TUV(x|a,b) = 0.5 - |a/(a+b) - 0.5|$ (see Eq. (2.2)). In the case of random sampling, the function is constant in the whole plane.

4.5 Implementation Using Kernel Density Classification

Implementing the active learning strategy presented in Section 4.4 requires distributional estimates $g_{2|x}$ for the posterior class probabilities at each unlabeled instance $x \in \mathcal{U}$. In this section, we derive such estimates for kernel density classification. To this end, we approximate the sampling distribution of the posterior point estimate.

Kernel density classification is a standard generative method. First, the priors $p(Y = i), i \in \{1,2\}$, and the densities $p(x|Y = i)$ are estimated for each class. Then, a point estimate for the posterior $p(Y = 2|x)$ can be obtained from Bayes' theorem:

$$\hat{p}(Y = 2|x) = \frac{\hat{p}(x|Y = 2)\hat{p}(Y = 2)}{\hat{p}(x|Y = 1)\hat{p}(Y = 1) + \hat{p}(x|Y = 2)\hat{p}(Y = 2)} \tag{4.14}$$

In kernel density classification, the class priors are usually estimated by the sample fractions $n_i/n = |\{(x,y) \in \mathcal{L} : y = i\}|/n$, the class densities by kernel density estimation:

$$\hat{p}(x|Y = i) = \frac{1}{n_i \det(H)} \sum_{x_j : y_j = i} \mathcal{K}\left(H^{-1}(x - x_j)\right) \tag{4.15}$$

where H is a nonsingular bandwidth matrix and $\mathcal{K}$ is a multivariate kernel function.

A standard assumption in kernel density estimation is that the training samples are drawn i.i.d. from (X, Y). For the following derivations, although not satisfied in active learning, this is also assumed for the labeled samples in $\mathcal{L}$. The key for deriving distributional estimates $g_{2|x}$ for posterior class probabilities is the insight that the density estimate given by Eq. (4.15) is of course not deterministic but depends on the randomness of the training data. This uncertainty in the density estimate $\hat{p}(x|Y = i)$ then carries over to uncertainty in the posterior point estimate $\hat{p}(Y = 2|x)$.

We start with modeling the uncertainty in density estimation with respect to the randomness of the training set $\mathcal{L}$. For a fixed point x, the expected value and the variance of the sampling distribution of $\hat{p}(x|Y = i)$ as defined in Eq. (4.15) can be approximated by [80, chap. 3]

$$\mathbf{E}_{\mathcal{L}}[\hat{p}(x|Y = i)] \approx p(x|Y = i) + \frac{1}{2}\mu_2(\mathcal{K})\text{tr}(H^T \mathcal{H}_{p(x|Y=i)} H) \tag{4.16}$$

and

$$\mathbf{V}_{\mathcal{L}}[\hat{p}(x|Y = i)] \approx \frac{1}{n_i \det(H)} \|\mathcal{K}\|_2^2 p(x|Y = i) \tag{4.17}$$

respectively, where $\mu_2(\mathcal{K}) := \int_{\mathbb{R}^d} \mathcal{K}(x)x^T x dx$, tr$(A)$ is the trace of the matrix A, $\mathcal{H}_{p(x|Y=i)}$ is the Hessian of $p(x|Y = i)$ at x and $\|\mathcal{K}\|_2^2 := \int_{\mathbb{R}^d}(\mathcal{K}(x))^2 dx$.

To estimate a full distribution instead of the moments only, we fit a Gamma distribution to the sampling distribution of $\hat{p}(x|Y = i)$ using the two moments in Eqs. (4.16) and (4.17). The exact choice of the distribution family is arbitrary here since the true distribution of $\hat{p}(x|Y = i)$ for finite sample sizes is not known [80, chap. 3]. However, there are four reasons that make the Gamma distribution a good candidate: (i) Both $\hat{p}(x|Y = i)$ and the Gamma distribution take values in $[0, \infty)$, (ii) the distribution family is sufficiently rich to approximate other distributions well, (iii) the Gamma distribution allows for the Bayesian treatment of the density estimation below, and (iv) the Gamma distribution model allows for an analytical derivation of the posterior uncertainty according to Eq. (4.22). As is common for computing confidence intervals [80, chap. 3] for $p(x|Y = i)$, we assume that the Hessian of $p(x|Y = i)$ in Eq. (4.16) vanishes at x, implying that $\hat{p}(x|Y = i)$ is an unbiased estimate of $p(x|Y = i)$, and we use a plug-in estimate for $p(x|Y = i)$ to estimate the variance in Eq. (4.17). The two parameters k and ϑ of the Gamma distribution can be easily determined by a moment estimator:

$$\begin{aligned} k\vartheta &\overset{!}{=} \hat{p}(x|Y = i) \\ k\vartheta^2 &\overset{!}{=} \frac{1}{n_i \det(H)} \hat{p}(x|Y = i) \|\mathcal{K}\|_2^2 \end{aligned}$$

We obtain

$$k = \frac{n_i \det(H)}{\|\mathcal{K}\|_2^2} \hat{p}(x|Y = i)$$

and

$$\vartheta = \frac{\|\mathcal{K}\|_2^2}{n_i \det(H)}$$

and thus

$$\hat{p}(x|Y=i) \dot{\sim} \Gamma\left(\frac{n_i \det(H)}{\|\mathcal{K}\|_2^2}\hat{p}(x|Y=i), \frac{\|\mathcal{K}\|_2^2}{n_i \det(H)}\right) \tag{4.18}$$

where $\dot{\sim}$ means "is approximately distributed as". Weighting the density estimate by the class priors, it follows from (4.18) and the scaling property of the Gamma distribution that

$$\frac{n_i}{n}\hat{p}(x|Y=i) \dot{\sim} \Gamma\left(\frac{n_i \det(H)}{\|\mathcal{K}\|_2^2}\hat{p}(x|Y=i), \frac{\|\mathcal{K}\|_2^2}{n \det(H)}\right) \tag{4.19}$$

Note the following summation property of the Gamma distribution: Let $X_i \sim \Gamma(k_i, \vartheta)$, $i = 1, \ldots, n$, be independent Gamma-distributed random variables; then,

$$\sum_{i=1}^{n} X_i \sim \Gamma\left(\sum_{i=1}^{n} k_i, \vartheta\right) \tag{4.20}$$

So, the first parameter of the distribution in (4.19) can be interpreted as the sum of the individual "density contributions" from the training data, i.e. $n_i\hat{p}(x|Y=i)$, weighted by $\det(H)/\|\mathcal{K}\|_2^2$. This insight be used now to motivate Eq. (4.21).

A general problem in AL is that there are only labels from one class at the beginning, at least after the first iteration of the process. Then, the first parameter of the Gamma distribution in (4.19) is zero and the distribution is not defined in this case. To overcome this problem, we assume a constant prior density throughout feature space for each class that is updated by the individual "density contributions" from the training data according to Property (4.20). I.e., we add a small constant $\delta > 0$, which yields

$$\frac{n_i}{n}\hat{p}(x|Y=i) \dot{\sim} \Gamma\left(\delta + \frac{n_i \det(H)}{\|\mathcal{K}\|_2^2}\hat{p}(x|Y=i), \frac{\|\mathcal{K}\|_2^2}{n \det(H)}\right) \tag{4.21}$$

It will turn out below that the uniform density prior in feature space corresponds to choosing a prior for the distributional estimate over the posterior class probabilities and thus that the exact choice of the additive constant in the first parameter of the Gamma distribution in Eq. (4.21) can be interpreted very well.

Next, we derive how the distribution statement in (4.21) for the weighted class densities carries over to an estimate for the sampling distribution of the posterior point estimate $\hat{p}(Y=2|x)$. It is shown in [130] that for two independent random variables $X_1 \sim \Gamma(k_1, \vartheta)$ and $X_2 \sim \Gamma(k_2, \vartheta)$,

$$\frac{X_2}{X_1 + X_2} \sim Beta(k_2, k_1) \tag{4.22}$$

Applying this result to Eqs. (4.14) and (4.21) yields an approximation to the sampling distribution of posterior point estimates in kernel density classification:

$$\mathcal{P}(\hat{p}(Y=2|x)) \approx Beta\left(\delta + \frac{n_2 \det(H)}{\|\mathcal{K}\|_2^2}\hat{p}(x|Y=2), \delta + \frac{n_1 \det(H)}{\|\mathcal{K}\|_2^2}\hat{p}(x|Y=1)\right) \tag{4.23}$$

This approximation is evaluated in Fig. 4.5, where it is compared to the empirical sampling distribution of $\hat{p}(Y=2|x)$ using a toy data example. Note that we revert to $Beta(\delta,\delta)$ if there are no labels in the neighborhood of x. Setting $\delta = 1/2$ corresponds to Jeffreys' uninformative prior [88] for the inference of the binomial proportion. The distribution in (4.23) can thus be interpreted as a posterior which is obtained from a Bayesian prior that has been updated by counting the labels (weighted by a kernel function) in the neighborhood of x.

Using the common Gaussian RBF kernel and setting $H := h \cdot I_d$ leads to

$$\hat{p}(x|Y=i) = \frac{1}{n_i h^d}\sum_{x_j : y_j = i} \frac{1}{\sqrt{(2\pi)^d}} e^{-\frac{1}{2}\left\|\frac{x-x_j}{h}\right\|^2} \tag{4.24}$$

for the density estimate in Eq. (4.15). Plugging this into Eq. (4.23) with $\delta = 1/2$ yields the principal result of this section:

$$\mathcal{P}(\hat{p}(Y=2|x)) \approx Beta\left(\underbrace{1/2 + 2^{\frac{d}{2}} \sum_{x_j : y_j = 2} e^{-\frac{1}{2}\left\|\frac{x-x_j}{h}\right\|^2}}_{=:a}, \underbrace{1/2 + 2^{\frac{d}{2}} \sum_{x_j : y_j = 1} e^{-\frac{1}{2}\left\|\frac{x-x_j}{h}\right\|^2}}_{=:b}\right) \tag{4.25}$$

where it is used that $\|\mathcal{K}\|_2^2 = 2^{-d}\pi^{-\frac{d}{2}}$. This distribution is used as distributional estimate for the implementation of the proposed active learning strategy, i.e. $g_{2|x} = g_{2|x}(q|a,b)$.

Finally, note an interesting connection to the derivations in Chapter 3. If we used the box or hypersphere kernel in Eq. (4.15), which corresponds to an ε-NN classifier, we would obtain

$$\hat{p}(x|Y=i) = \frac{1}{n_i h^d}\sum_{x_j : y_j = i} c_d^{-1}\mathbf{1}\{\|x_j - x\| \leq h\}$$

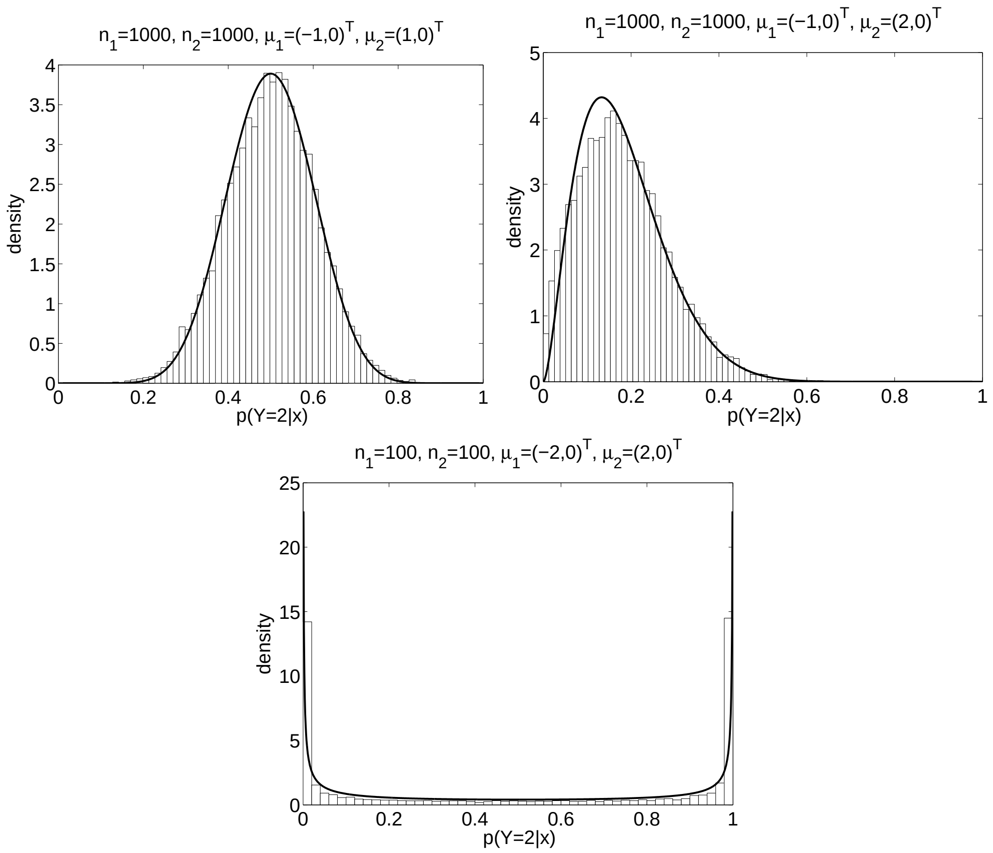


Figure 4.5: Comparison of the empirical sampling distribution of $\hat{p}(Y = 2|x)$ (represented by the histograms) and the estimated sampling distribution in Eq. (4.23) resulting from the Gamma distribution model (solid lines). In each panel, instances of classes 1 and 2 are sampled from a two-dimensional Gaussian distribution with mean μ_1 and μ_2, respectively, and equal unit covariance matrix. The number of samples is equal to n_1 and n_2, respectively, and $x = (0, 0)^T$. The histograms are computed from $10{,}000$ repetitions, i.e. $10{,}000$ different realizations of the training set. For each realization, $\hat{p}(Y = 2|x)$ is obtained using a Gaussian RBF kernel with $H = 0.1 \cdot I_d$ for the density estimate of each of the two classes (see also Eq. (4.24)). For the approximation of the sampling distribution according to Eq. (4.23), the two density estimates are replaced by their known true values and the constant δ has been set to 0 to make the simulations independent from the prior.

for the density estimate, where c_d is the volume of the d-dimensional unit ball, and

$$\begin{aligned} &\mathcal{P}(\hat{p}(Y = 2|x)) \\ &\approx Beta\left(1/2 + \sum_{x_j : y_j = 2} \mathbf{1}\{\|x_j - x\| \le h\}, 1/2 + \sum_{x_j : y_j = 1} \mathbf{1}\{\|x_j - x\| \le h\}\right) \end{aligned} \tag{4.26}$$

for the sampling distribution, where it is used that $\|\mathcal{K}\|_2^2 = c_d^{-1}$. I.e., if the labels in a neighborhood of x are weighted equally (corresponding to the employed kernel), the parameters of the Beta are given by 1/2 plus the number of labels of the respective classes in the neighborhood of x. This is the posterior distribution of a standard Bayesian inference model for the success parameter of the Binomial distribution (see e.g. [20, chap. 2]) using Jeffreys' prior [88] $Beta(1/2, 1/2)$ and has already been derived in Eq. (3.3).

4.6 Results

In this section, we present experimental results for the DEAL strategy. Using a toy data example, we first demonstrate that DEAL trades off exploration and exploitation in a natural way (Section 4.6.1). Then in Sections 4.6.2 and 4.6.3, using real world data sets from the UCI repository [9] and Caltech-4 [60], we compare the proposed approach to random sampling and uncertainty sampling. Finally, we compare DEAL to LSS (look-ahead selective sampling) in Section 4.6.4, a method previously proposed in [111].

For the implementation of kernel density classification, an isotropic Gaussian kernel and the normal reference rule [159, chap. 6] for the kernel width is used. The density $\hat{p}(x)$ in Eq. (4.13) is estimated by kernel density estimation with the same kernel type and width.

In all experiments, we always start with an empty set $\mathcal{L}$ of labeled points. In case of uncertainty sampling, the first query points are selected randomly until there is at least one label for each class. In case of DEAL, the first label is automatically requested for the point with the highest density estimate; afterwards, the strategy can be applied even if there are labels of one class only. If not stated otherwise, all results are obtained from 2 × 5-fold cross validation.

4.6.1 XOR Problem

Uncertainty sampling is known to query too many labels in regions with low density [122] and it neglects exploration in favor of exploitation. We demonstrate the latter using the well-known two-dimensional XOR problem and show that DEAL overcomes this problem. Here, each class is a mixture of two equally weighted Gaussians with unit covariance. The mean values are $(-3, -3)$ and $(3, 3)$ for class 1 and $(-3, 3)$ and $(3, -3)$ for class 2. The number of instances is 150 for each class resulting in a (initially unlabeled) training set of 240 and a test set of 60 samples for each run. The realizations are normalized to unit variance in each dimension.

Typical patterns for label query order of the two different AL strategies (both resulting from the same training set) are plotted in Fig. 4.6. It can be observed that DEAL explores feature space thoroughly, whereas uncertainty sampling completely overlooks the class

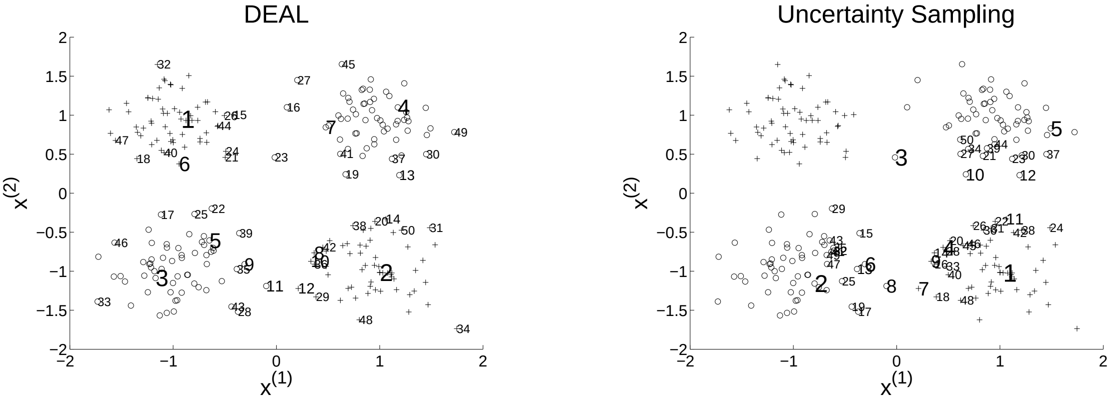


Figure 4.6: Label query order of the first 50 labels for two different AL strategies, implemented with kernel density classification. Note that the data is normalized to unit variance in each dimension. Uncertainty sampling completely overlooks the relevance of the observations in the second quadrant of the coordinate system as labels are queried only at the decision boundary. In contrast, the proposed strategy thoroughly explores the feature space. Note in particular that all 4 clusters have been visited after 4 label queries and that the corresponding query points are close to the cluster centers due to considering density information. Labels 5–8 are again distributed over all 4 clusters, this time at some distance to the previous labels in direction of the decision boundary.

2 realizations in the second quadrant of the coordinate system due to too greedy a label query at the decision boundary. This problem could not even be solved by incorporating density information: The predictions for the posterior class probability $p(Y = 2|x)$ of unlabeled points are all equal to 0 or very close to 0 for the unlabeled training points in the second quadrant. In contrast, the proposed approach additionally takes the number of labeled points in the neighborhood into account which results in a more systematic exploration of feature space. Nevertheless, instances close to the decision boundary are labeled as well within the first queries. The averaged learning curves for the XOR problem with respect to accuracy are plotted in Fig. 4.7. Note in particular that, for the reasons discussed above, uncertainty sampling leads to even worse results than random sampling.

4.6.2 UCI Data Sets

Above, we demonstrated the advantages of DEAL over uncertainty sampling using a simple toy data example. Now, we show that these considerations indeed have an impact on the performance in real world problems. To this end, we compare the proposed approach to uncertainty and random sampling on 32 data sets from the UCI data base. Each of the different data sets is preprocessed as follows: (i) Categorical variables with

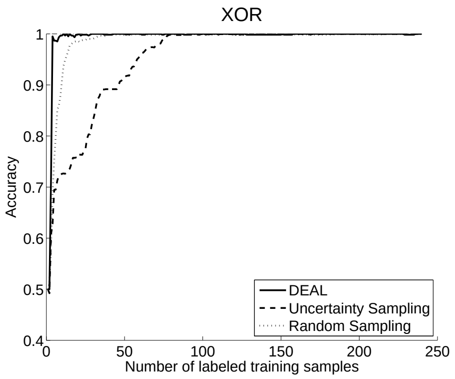


Figure 4.7: Learning curves for the XOR problem with respect to accuracy

more than two outcomes are replaced by dummy variables, (ii) missing values in categorical variables are treated as a separate outcome, (iii) missing values in continuous inputs are replaced by the respective mean, and (iv) the data is normalized to unit variance in each dimension. If a data set has more than two classes, the classes are joined to create two-class problems in a way such that the new classes are approximately equally abundant.

As kernel density estimation is known to be problematic in high dimensions [159, chap. 7] and inappropriate for discrete or binary features, the data is additionally preprocessed by principal component analysis. For automatically determining the number of principal components to be used, we applied a scheme presented e.g. in [191][6] and [81, chap. 14]. To avoid oversimplified data sets, the minimum number of components is set to two.

To the best of our knowledge, there is no standard method for measuring AL performance. We propose to compare the different strategies by averaging over the performance after each label query or, equivalently, by computing the area under the learning curves. This measure honors both initial steepness of the learning curve and early convergence of the performance to a high level. As we compare only the relative performance of different strategies for the same classification algorithm, the measure is equivalent to the one proposed in [12] and also used in [156] (called “deficiency”). Example learning curves for the data sets “Iris” and “Optdigits” are shown in Fig. 4.8. The results for all data sets are presented in Table 4.2, The corresponding learning curves are shown in Appendix B of this chapter in Section 4.9.

It can be seen that the proposed strategy performs better than uncertainty sampling and random sampling for most of the data sets. We compare the different strategies as recommended in [45]. The Friedman test yields $p = 0.001$ for the hypothesis of equal

[6] In [191], it is called the “resampling scheme via permutation”.

Dataset	DEAL	Unc. Sampl.	RS	# samples	d
Anneal	0.924 (1)	0.880 (3)	0.898 (2)	898	17
Audiology	0.753 (1)	0.726 (3)	0.736 (2)	226	9
Autos	0.704 (1)	0.672 (3)	0.675 (2)	205	14
Balance-Scale	0.733 (2)	0.730 (3)	0.734 (1)	625	2
Breast-Cancer	0.649 (3)	0.662 (1)	0.652 (2)	286	16
Breast-W	0.963 (1)	0.958 (2)	0.952 (3)	699	2
Dermatology	0.987 (1)	0.985 (2)	0.973 (3)	366	4
Diabetes	0.708 (1)	0.696 (3)	0.699 (2)	768	2
Ecoli	0.886 (1)	0.870 (3)	0.875 (2)	336	3
Glass	0.723 (1)	0.715 (2)	0.686 (3)	214	4
Heart-C	0.757 (2)	0.759 (1)	0.733 (3)	303	8
Hepatitis	0.826 (2)	0.818 (3)	0.828 (1)	155	7
Hypothyroid	0.928 (1)	0.920 (2)	0.913 (3)	3772	11
Ionosphere	0.903 (1)	0.882 (3)	0.884 (2)	351	5
Iris	0.990 (1)	0.982 (2)	0.981 (3)	150	2
Led24	0.689 (1)	0.674 (3)	0.689 (2)	1000	2
Letters	0.679 (1)	0.655 (3)	0.658 (2)	20000	5
Liver	0.556 (1)	0.544 (2)	0.542 (3)	345	2
Lymph	0.710 (2)	0.732 (1)	0.702 (3)	148	9
Optdigits	0.941 (1)	0.907 (3)	0.913 (2)	5620	18
Pendigits	0.927 (1)	0.903 (2)	0.880 (3)	7494	5
Primary-Tumor	0.670 (2)	0.674 (1)	0.659 (3)	339	9
Satimage	0.951 (2)	0.957 (1)	0.919 (3)	6435	3
Segment	0.874 (1)	0.774 (3)	0.845 (2)	2310	3
Sonar	0.776 (1)	0.765 (3)	0.773 (2)	208	8
Soybean	0.902 (1)	0.899 (2)	0.884 (3)	683	20
Vehicle	0.800 (1)	0.790 (2)	0.781 (3)	846	4
Vote	0.882 (1)	0.877 (2)	0.874 (3)	435	8
Vowel	0.753 (1)	0.604 (3)	0.725 (2)	990	16
Waveform	0.874 (2)	0.875 (1)	0.864 (3)	5000	2
Wine	0.948 (1)	0.942 (2)	0.935 (3)	178	3
Yeast	0.727 (1)	0.723 (2)	0.718 (3)	1484	2
Mean Rank	1.281	2.250	2.469		

Table 4.2: Average accuracy of different AL strategies, implemented with kernel density classification. Numbers in brackets refer to the Friedman test; the number of samples comprises all folds, i.e. training *and* test set, and d is the feature space dimension after having applied principal component analysis.

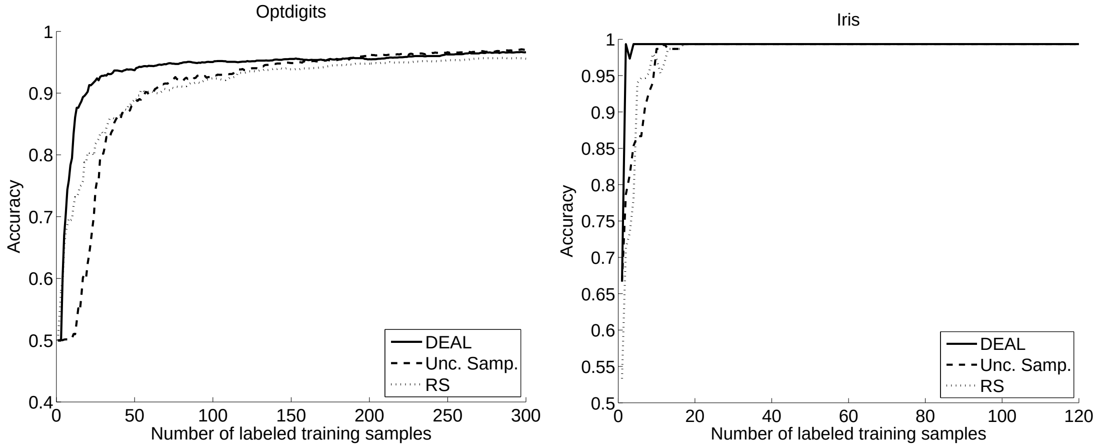


Figure 4.8: Learning curves of the three different AL strategies with respect to accuracy for data sets "Optdigits" and "Iris". Learning curves for the remaining UCI data sets are shown in Appendix B of this chapter in Section 4.9. The results for "Iris" are typical for data sets with well-separated classes. Uncertainty sampling and random sampling achieve good classification with relatively few labels, but DEAL needs even fewer labels. The data set "Optdigits" is an example where sampling at the decision boundary only more or less completely fails at the beginning of the AL process, yielding results that are even worse than those for random sampling. In contrast, DEAL is very efficient from the beginning.

performance of all strategies. For comparing all classifiers to each other, we use the two-tailed Nemenyi test. Its critical difference for the 0.01 significance level is equal to 0.728. This means that DEAL performs significantly best and that the performance of uncertainty sampling does not differ significantly from random sampling (the critical difference for the 0.1 significance level is 0.513).

4.6.3 Caltech-4

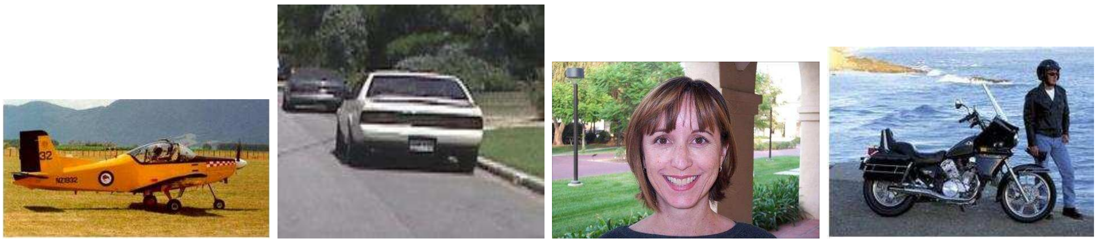


Figure 4.9: Example images of the 4 object categories of Caltech-4. From left to right: airplane, car, face, motorbike.

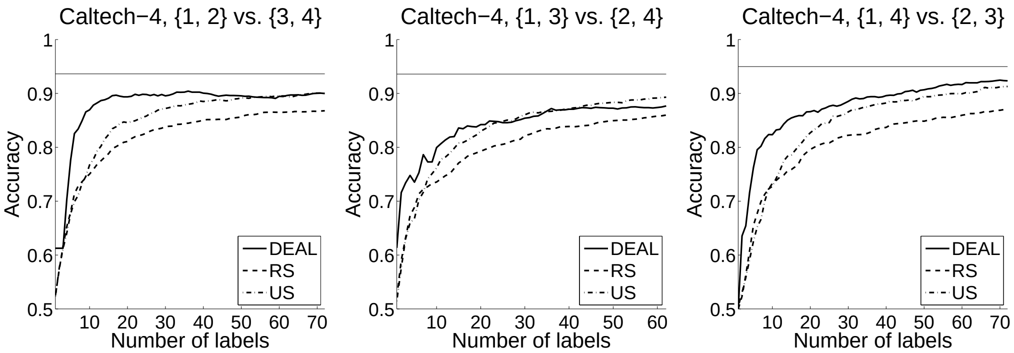


Figure 4.10: Learning curves for three possible groupings of the 4 categories. DEAL performs best in all cases and US second best. The top horizontal line is the asymptotic accuracy of the classifier, with all training data labeled (estimated by 10-fold CV).

Grouping	RS	US	DEAL
$\{1,2\}$ vs. $\{3,4\}$	0.818	*0.846*	**0.877**
$\{1,3\}$ vs. $\{2,4\}$	0.799	*0.829*	**0.840**
$\{1,4\}$ vs. $\{2,3\}$	0.803	*0.836*	**0.872**
Mean Rank	3.000	2.000	1.000

Table 4.3: Average accuracy of the compared AL strategies for 3 different groupings of the Caltech-4 data set with preprocessing as described in text. The best and second best method are indicated using bold font and italics, respectively.

Caltech-4 is a well established standard benchmark for object categorization [60] and has also been used in AL [93]. This dataset consists of 4 different image groups: airplanes (category 1; 800 images), rear views of cars (2; 1155), frontal faces (3; 435) and motorbikes (4; 798). Fig. 4.9 shows one example from each category. We represent the images by the “Color and Edge Directivity Descriptor” (CEDD) [34]. The resulting 144-dimensional features were then projected to the 17 leading principal components using the same method as in Section 4.6.2. To create challenging two-class problems with convoluted decision boundaries, we grouped the 4 categories in three possible ways.

The resulting learning curves are shown in Fig. 4.10, based on 10-fold CV with 5 repetitions. Table 4.3 compares the performances based on the area under the learning curve. It shows that, for all groupings, DEAL performs best, uncertainty sampling second best and random sampling worst.

4.6.4 USPS Zip Data

In this section, we compare DEAL to LSS as proposed in [111]. To this end, we create challenging classification problems with many data clusters by using various dichotomies of the digits from the USPS zip corpus [102]. The groupings are (1) $\{0, 1, 2, 3, 4\}$ vs. $\{5, 6, 7, 8, 9\}$ (0-4 vs. rest), (2) $\{1, 2, 3, 4, 5\}$ vs. $\{6, 7, 8, 9, 0\}$ (1-5 vs. rest), (3) $\{1, 3, 5, 7, 9\}$ vs. $\{0, 2, 4, 6, 8\}$ (odd vs. even), (4) $\{0, 1, 7, 8, 9\}$ vs. $\{2, 3, 4, 5, 6\}$ (all digits contained in my date of birth vs. rest), (5) $\{1, 3, 4, 5, 9\}$ vs. $\{2, 6, 7, 8, 0\}$ (the first five different digits of π vs. rest). Similar to the previous sections, the handwritten digits from USPS zip corpus are projected onto their first principal components to reduce the dimensionality of the digitized images (see e.g. [81, chap. 14]). For automatically determining the number of principal components to be used, we apply the same method as before. This yields $d = 39$. For a second experiment, we determine the number of features by visual inspection of the eigenvalues. This yields $d = 12$. The size of the unlabeled pool is 7291, the test set comprises 2007 samples. The results, averaged over three runs, are shown in Figs. 4.11–4.15.

The underlying classifier of LSS is weighted (by distance) 2-nearest-neighbor (2-NN). If $\theta = 1/2$, the class assignment is equivalent to 1-NN. Therefore, 1-NN is chosen as the underlying classifier for random sampling in the context of LSS. It can be observed from the two different learning curves of random sampling (1-NN for LSS and kernel density classification for DEAL) that the performance of the underlying classifier with respect to the USPS Zip Data is approximately equal. Sometimes, kernel performs slightly better, sometimes 1-NN. Hence, performance differences can be attributed to the AL strategy itself. It can be observed that DEAL generally performs better for all five partitions of the data in both 12 and 39 dimensions. Very rarely, the learning curves of the strategies intersect, e.g., LSS performs better than DEAL with grouping (5) (Fig. 4.15) in 12 dimensions with about 50 labels.

4.7 Conclusions

In this chapter, we have derived a novel two-class AL strategy, which considers not only density information and the distance to the decision boundary when selecting an instance to be labeled, but also the number of labeled points in the neighborhood. All this information is taken into account by requiring that the underlying classifier provide a distributional estimate for each unlabeled point leading to a natural definition of the training utility value.

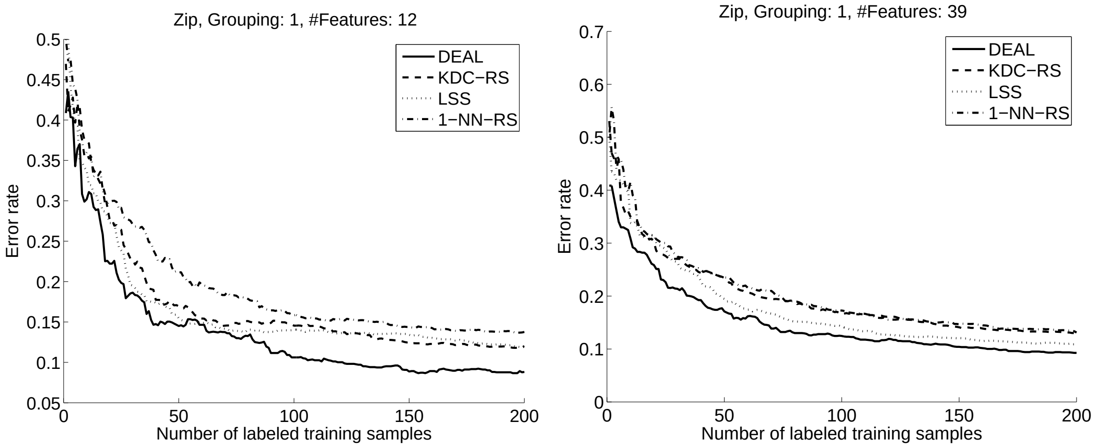


Figure 4.11: Comparison of DEAL with Look-Ahead Selective Sampling, grouping “0-4 vs. rest”. DEAL has steeper learning curves than LSS. Since the underlying classifiers (kernel density classification for DEAL and 1-NN for LSS) show approximately equal classification performance for random sampling, the difference of the AL performance can be attributed to the AL strategy itself.

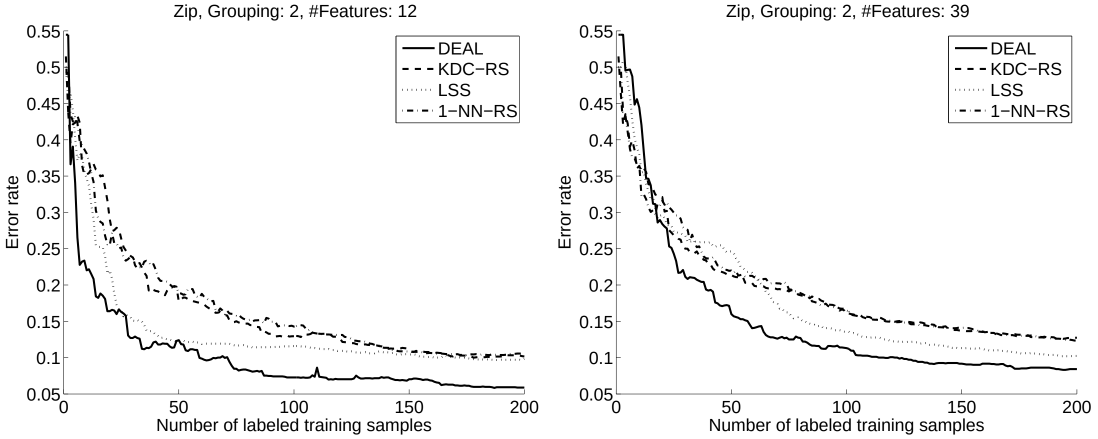


Figure 4.12: Comparison of DEAL with Look-Ahead Selective Sampling, grouping “1-5 vs. rest”. Further explanations can be found in the text and in the caption of Fig. 4.11.

To the best of our knowledge, this is the first generic approach which considers the number of labeled points in the neighborhood of a yet unlabeled point in linear time complexity (for a single request) with respect to the total number of unlabeled points.

The proposed implementation of our strategy is a counter example to the claim made in [54] that a single model cannot be used to estimate a second-order uncertainty: The

> “type of uncertainty regarding the identity of the appropriate classification, is different than uncertainty regarding the correctness of the classifi-

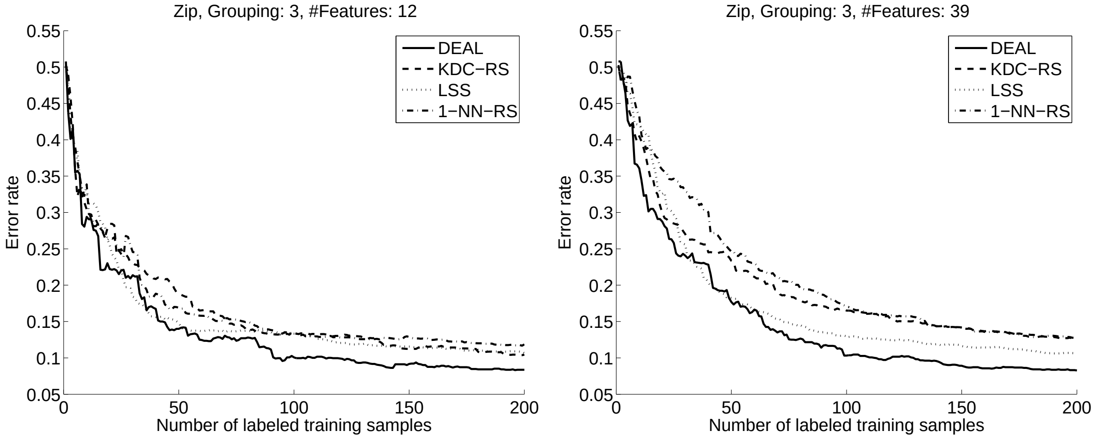


Figure 4.13: Comparison of DEAL with Look-Ahead Selective Sampling, grouping “odd vs. even”. Further explanations can be found in the text and in the caption of Fig. 4.11.

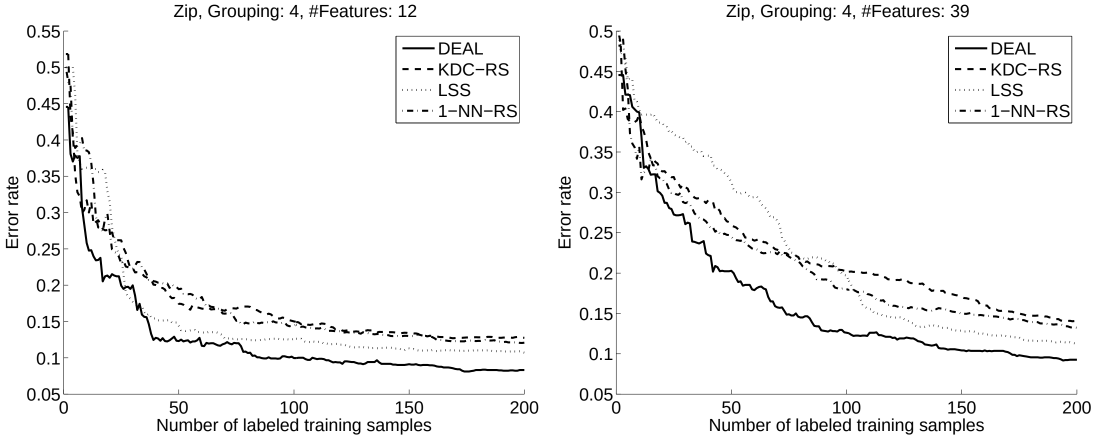


Figure 4.14: Comparison of DEAL with Look-Ahead Selective Sampling, grouping $\{0, 1, 7, 8, 9\}$ vs. $\{2, 3, 4, 5, 6\}$. Further explanations can be found in the text and in the caption of Fig. 4.11.

> cation itself. For example, sufficient statistics may yield an accurate 0.51 probability estimate for a class c in a given example, making it certain that c is the *appropriate* classification[7]. However, the certainty that c is the *correct* classification is low, since there is a 0.49 chance that c is the wrong class for the example. A single model can be used to estimate only the second type of uncertainty, which does not correlate directly with the utility of additional training.”

The proposed AL approach significantly outperforms uncertainty and random sam-

[7] “Appropriate” classification refers here to the right class assignment in the sense of decision theory.

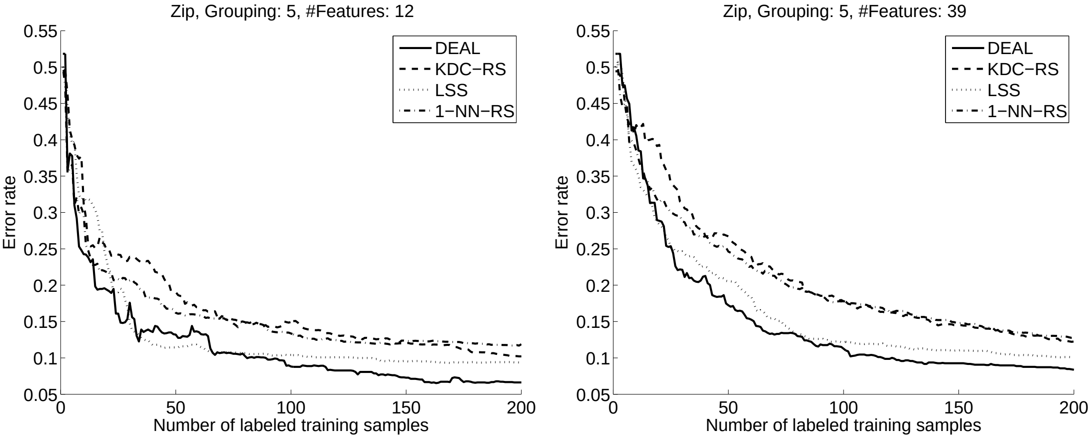


Figure 4.15: Comparison of DEAL with Look-Ahead Selective Sampling, grouping $\{1, 3, 4, 5, 9\}$ vs. $\{2, 6, 7, 8, 0\}$. Further explanations can be found in the text and in the caption of Fig. 4.11.

pling. This has been shown by a comprehensive evaluation of the proposed strategy. Moreover, we have shown that it performs better than "Look-Ahead Selective Sampling" for the demanding USPS Zip data. In [151], it is shown that the strategy also outperforms "error reduction sampling" [153][8]. Note that, in general, one cannot expect that one strategy is best on all data sets: Of course, strongly favoring exploitation over exploration (like in uncertainty sampling) can be a good strategy if the distribution of the data is sufficiently simple.

The observation in Table 4.2 that random sampling sometimes outperforms uncertainty sampling is consistent with other AL evaluations. [156] compare seven different AL strategies (including uncertainty sampling and query by committee), implemented using logistic regression; none of these is always better than random sampling although only seven non-artificial data sets are considered. In [75], random sampling is the *best* approach on two out of nine data sets compared to four other (batch mode) AL strategies. And in [164], one specific implementation of uncertainty sampling is the only (sequence labeling) AL strategy out of 15 which outperforms random sampling on all eight data sets. However, this "successful" strategy yields poor average results compared to the others.[9]

[8] This is a self-citation. The corresponding results are not presented in this chapter because the "error reduction sampling" experiments have been implemented by Kevin Kunzmann.

[9] The last two sentences are not contradictory!

4.8 Appendix A: Proofs

First, we prove Eq. (4.9) in Section 4.4, which states that $R_x(\hat{p}(Y=2|x)) \geq \hat{R}_x(g_{2|x})$, i.e. that the expected loss of a distribution is smaller or equal to the expected loss of the point estimate given by the mean of this distribution and thus, the training utility value is always non-negative. The inequality immediately follows from Proposition 4.2 which also states when equality holds. The latter proves Eq. (4.10). As the distribution of X is not assumed to be continuous in Proposition 4.2, but can be discrete or mixed continuous/discrete, we use notation from measure theory to perform integration with respect to an arbitrary probability measure.

Proposition 4.2. *Let X be a real-valued random variable with $\mathbf{P}(0 \leq X \leq 1) = 1$. Further, let $L_{12}, L_{21} > 0$ and*

$$\theta = \frac{L_{12}}{L_{12} + L_{21}} \in (0,1)$$

Then,

$$\begin{aligned}&\int L_{21} X\, \mathbf{1}\{X \leq \theta\} d\mathbf{P}^X + \int L_{12}(1-X)\, \mathbf{1}\{X > \theta\} d\mathbf{P}^X \\ \leq\, & L_{21}\mathbf{E}(X)\, \mathbf{1}\{\mathbf{E}(X) \leq \theta\} + L_{12}(1-\mathbf{E}(X))\, \mathbf{1}\{\mathbf{E}(X) > \theta\}\end{aligned}$$

Equality holds iff $\mathbf{P}(X \leq \theta) = 1$ or $\mathbf{P}(X \geq \theta) = 1$.

Proof. First of all, $\mathbf{E}(X)$ exists since $\int |X| d\mathbf{P}^X \leq \int d\mathbf{P}^X = 1 < \infty$.

Let $\mathbf{E}(X) \leq \theta$. Then,

$$\begin{aligned}&\int L_{21} X\, \mathbf{1}\{X \leq \theta\} d\mathbf{P}^X + \int L_{12}(1-X)\, \mathbf{1}\{X > \theta\} d\mathbf{P}^X \\ =&\int_{X \leq \theta} L_{21} X d\mathbf{P}^X + \int_{X > \theta} L_{21} \frac{\theta}{1-\theta}(1-X) d\mathbf{P}^X \\ \leq\, & L_{21} \int_{X \leq \theta} X d\mathbf{P}^X + L_{21} \int_{X > \theta} \frac{X}{1-X}(1-X) d\mathbf{P}^X \\ =\, & L_{21}\mathbf{E}(X)\end{aligned}$$

Equality holds iff

$$\int_{X>\theta} \frac{\theta}{1-\theta}(1-X) d\mathbf{P}^X = \int_{X>\theta} \frac{X}{1-X}(1-X) d\mathbf{P}^X \qquad \Leftrightarrow \qquad \mathbf{P}(X > \theta) = 0$$

Now, let $\mathbf{E}(X) > \theta$. Then,

$$\begin{aligned}
&\int L_{21} X\,\mathbf{1}\{X \leq \theta\} d\mathbf{P}^X + \int L_{12}(1-X)\,\mathbf{1}\{X > \theta\} d\mathbf{P}^X \\
=& \int_{X \leq \theta} L_{12} \frac{1-\theta}{\theta} X d\mathbf{P}^X + \int_{X > \theta} L_{12}(1-X) d\mathbf{P}^X \\
\leq& L_{12} \int_{X \leq \theta} \frac{1-X}{X} X d\mathbf{P}^X + L_{12} \int_{X > \theta} (1-X) d\mathbf{P}^X \\
=& L_{12}(1 - \mathbf{E}(X))
\end{aligned}$$

Equality holds iff

$$\int_{X \leq \theta} \frac{1-\theta}{\theta} X d\mathbf{P}^X = \int_{X \leq \theta} \frac{1-X}{X} X d\mathbf{P}^X \qquad \Leftrightarrow \qquad \mathbf{P}(X < \theta) = 0$$

□

Now, we perform the calculations to obtain term (4.12) from term (4.11) in Section 4.4. Let $a, b > 0$. The Beta function is defined as

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

(see e.g. [2, chap. 6.2]) where $\Gamma(\cdot)$ is the well-known Gamma function. The incomplete Beta function is defined as

$$I_\theta(a, b) = \frac{1}{B(a, b)} \int_0^\theta q^{a-1}(1-q)^{b-1} dq$$

(see e.g. [2, chap. 26.5]) where $\theta \in [0, 1]$. Using these definitions, we have

$$\begin{aligned}
&\hat{p}(x) \int_0^1 [\mathbf{1}\{q \leq \theta\} q L_{21} + \mathbf{1}\{q > \theta\}(1-q) L_{12}] p_{2|x}(q|a, b) dq \\
=& \hat{p}(x) \left[\int_0^\theta \frac{L_{21}}{B(a, b)} q^a (1-q)^{b-1} dq + \int_\theta^1 \frac{L_{12}}{B(a, b)} q^{a-1} (1-q)^b dq \right] \\
=& \hat{p}(x) \left[\frac{B(a+1, b) L_{21}}{B(a, b)} \int_0^\theta \frac{1}{B(a+1, b)} q^a (1-q)^{b-1} dq \right. \\
&\left. + \frac{B(a, b+1) L_{12}}{B(a, b)} \int_0^\theta \frac{1}{B(a, b+1)} q^{a+1} (1-q)^b dq \right] \\
=& \hat{p}(x) \left[\frac{a L_{21}}{a+b} I_\theta(a+1, b) + \frac{b L_{12}}{a+b} I_{1-\theta}(b+1, a) \right]
\end{aligned}$$

4.9 Appendix B: Learning Curves

Here, we present the learning curves that yielded the results in Table 4.2. For each data set, the proposed AL strategy is compared to uncertainty and random sampling.

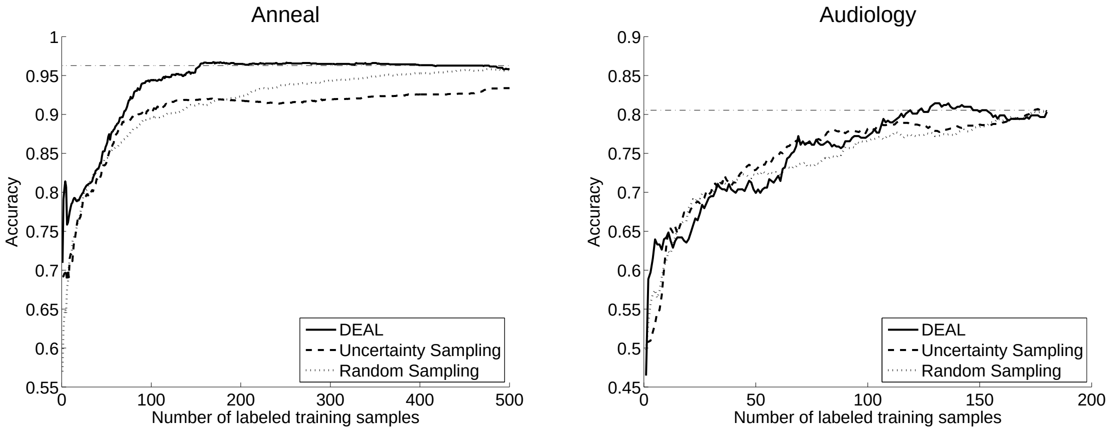


Figure 4.16: Learning curves for data sets “Anneal” and “Audiology”

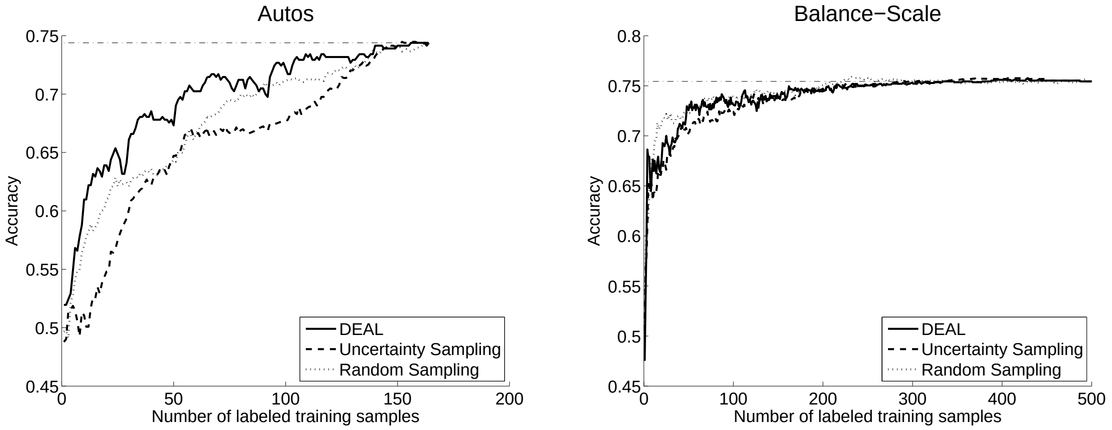


Figure 4.17: Learning curves for data sets “Autos” and “Balance-Scale”

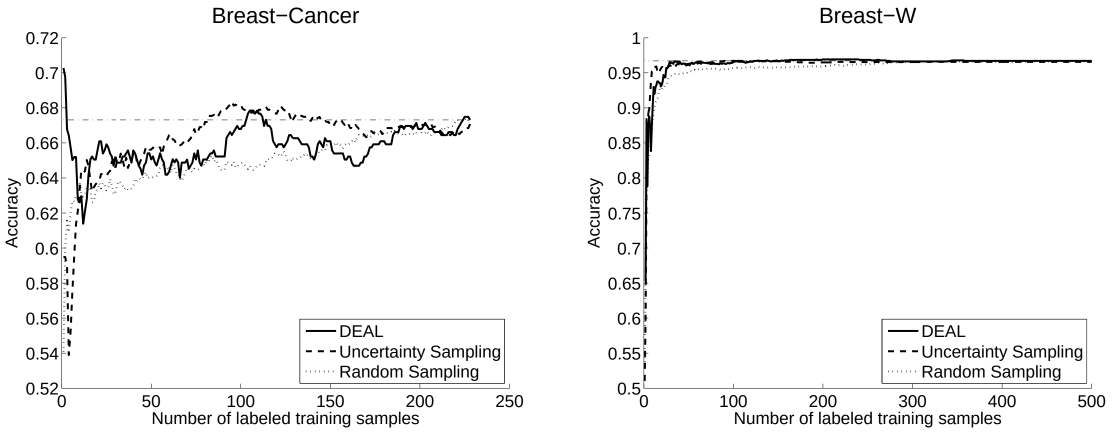


Figure 4.18: Learning curves for data sets “Breast-Cancer” and “Breast-W”

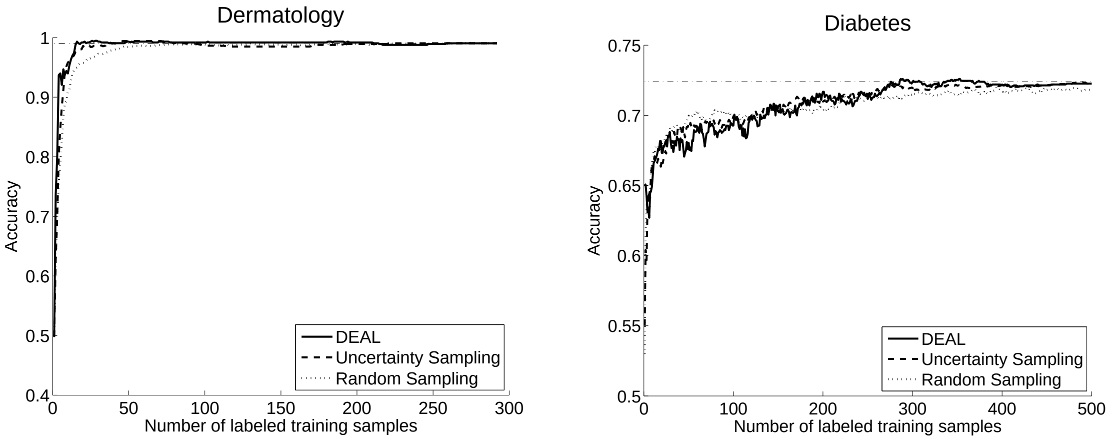


Figure 4.19: Learning curves for data sets “Dermatology” and “Diabetes”

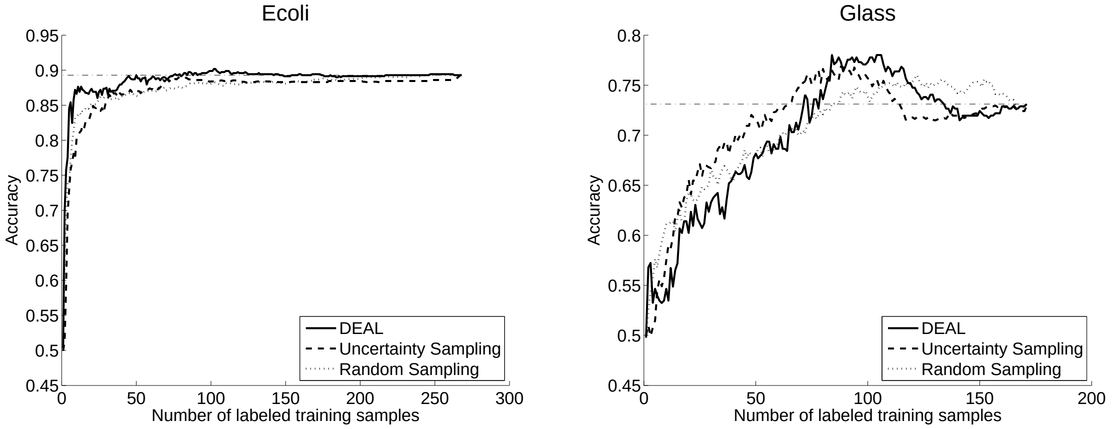


Figure 4.20: Learning curves for data sets “Ecoli” and “Glass”

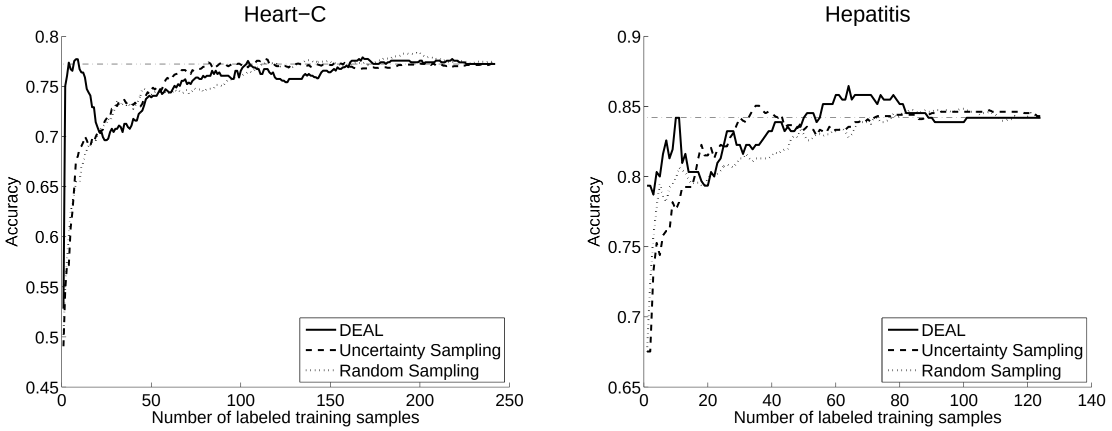


Figure 4.21: Learning curves for data sets “Heart-C” and “Hepatitis”

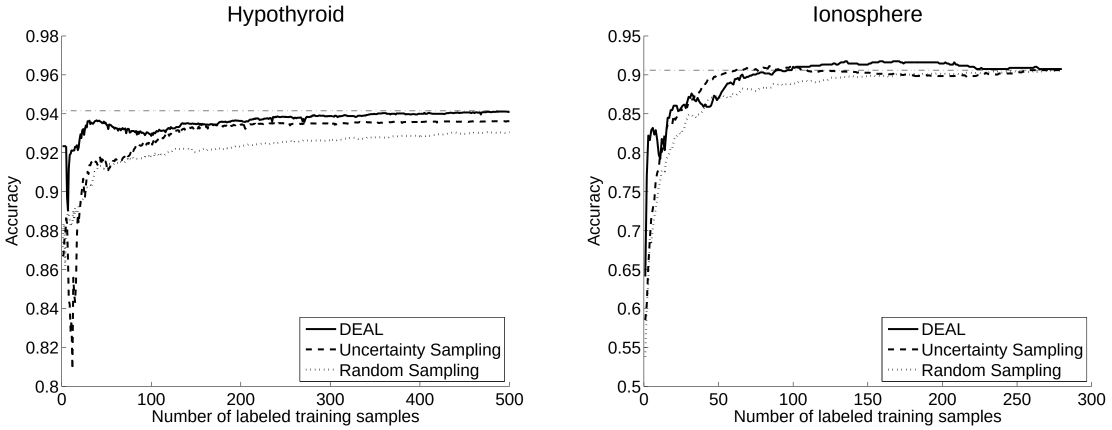


Figure 4.22: Learning curves for data sets “Hypothyroid” and “Ionosphere”

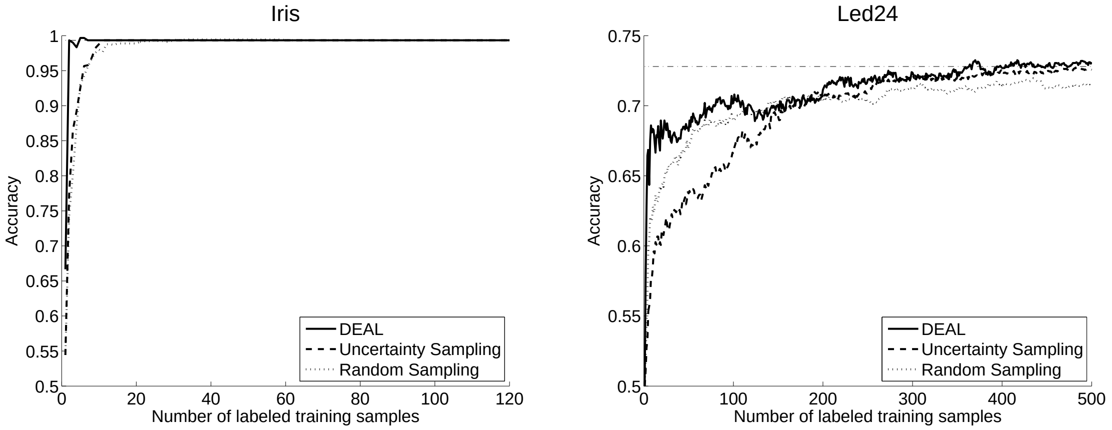


Figure 4.23: Learning curves for data sets “Iris” and “Led24”

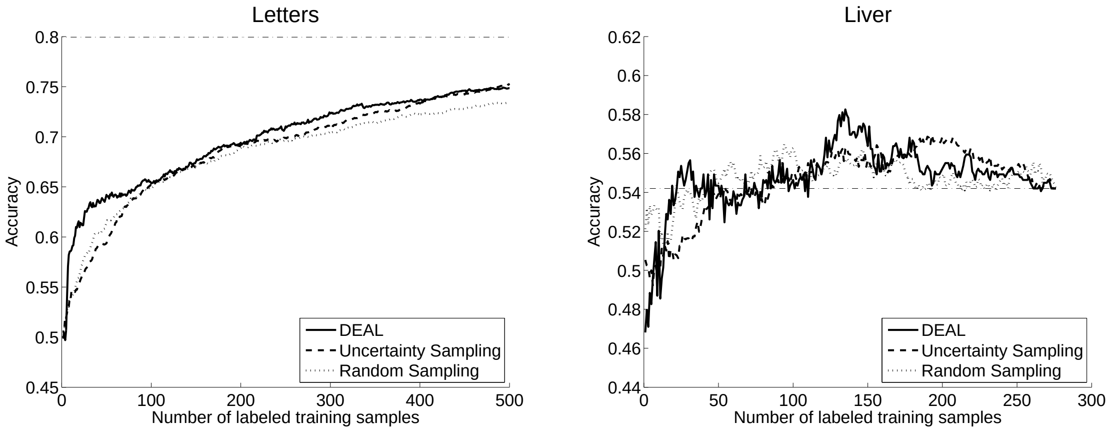


Figure 4.24: Learning curves for data sets “Letters” and “Liver”

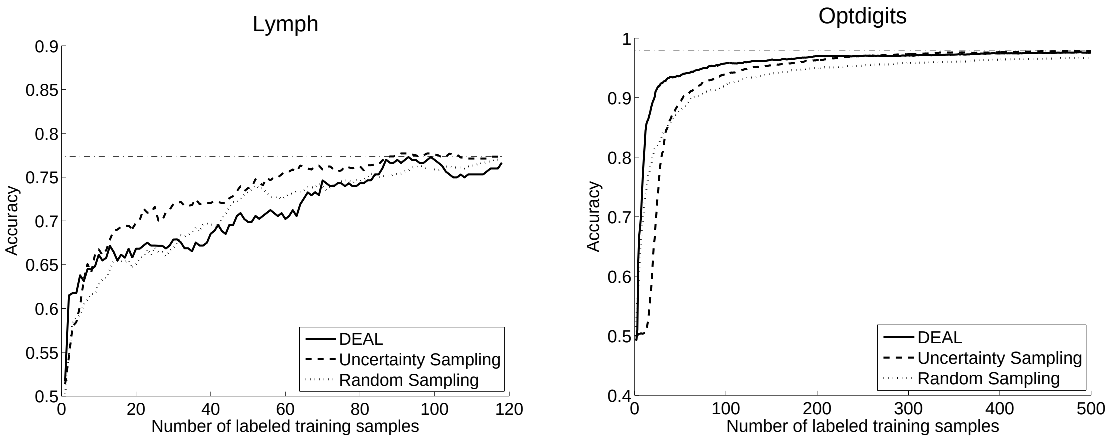


Figure 4.25: Learning curves for data sets “Lymph” and “Optdigits”

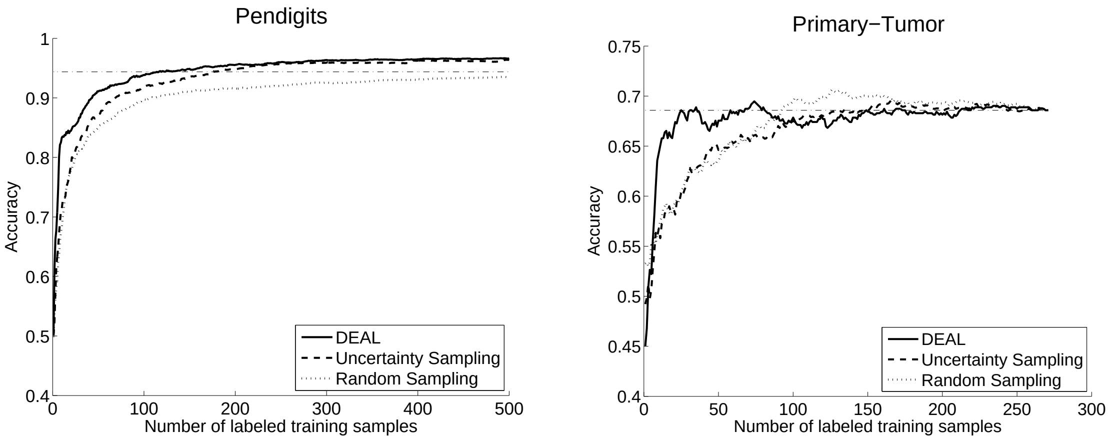


Figure 4.26: Learning curves for data sets “Pendigits” and “Primary-Tumor”

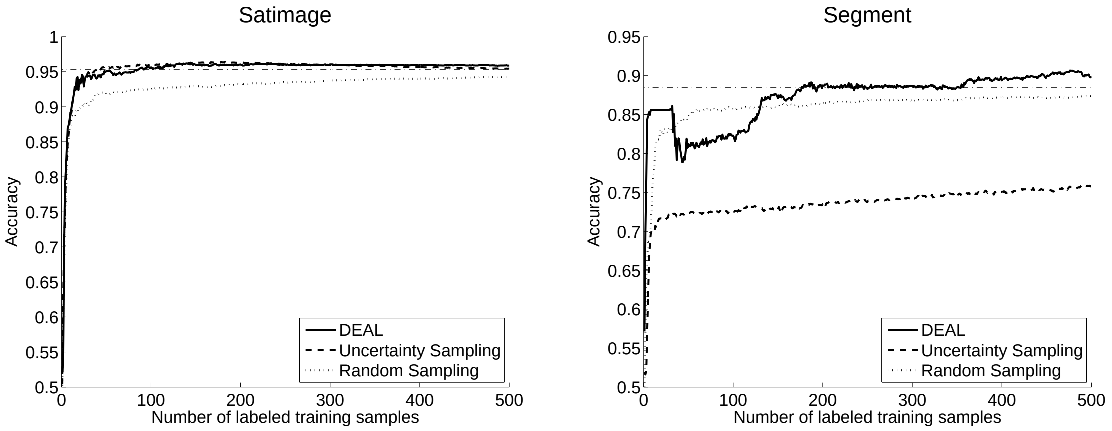


Figure 4.27: Learning curves for data sets “Satimage” and “Segment”

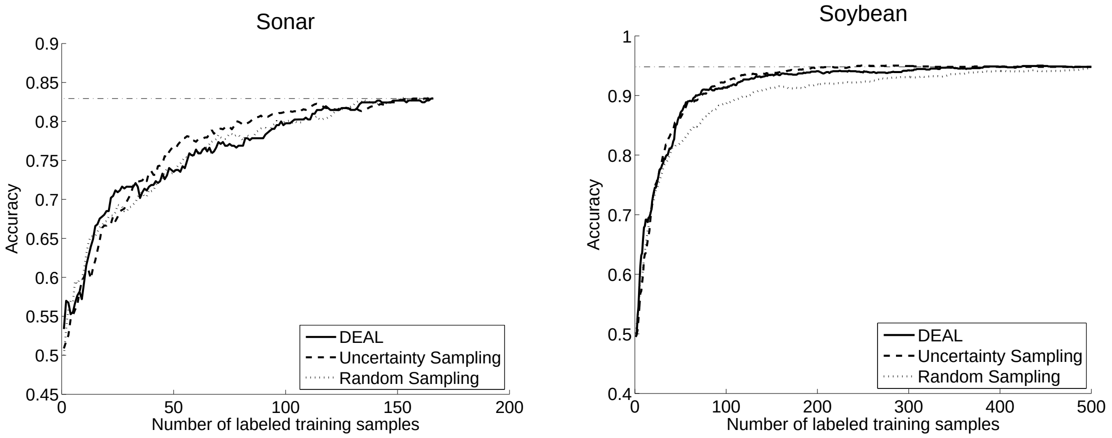


Figure 4.28: Learning curves for data sets “Sonar” and “Soybean”

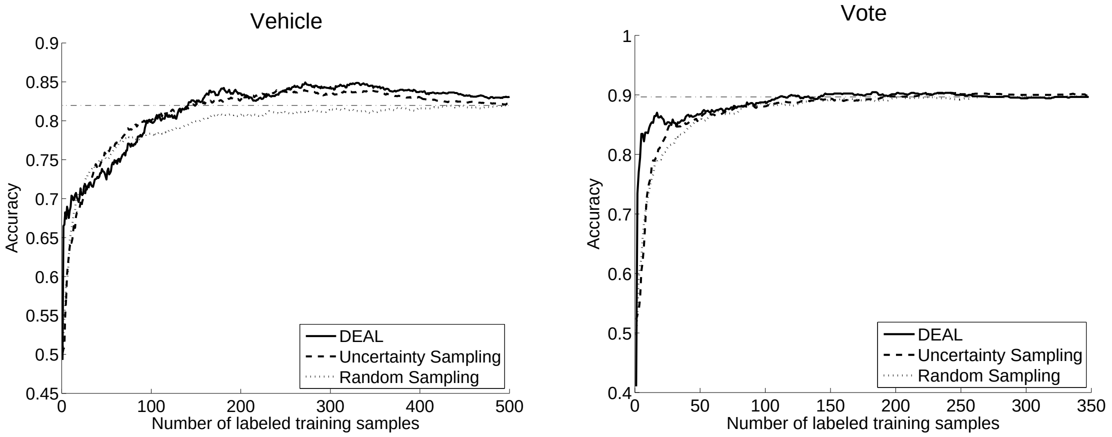


Figure 4.29: Learning curves for data sets “Vehicle” and “Vote”

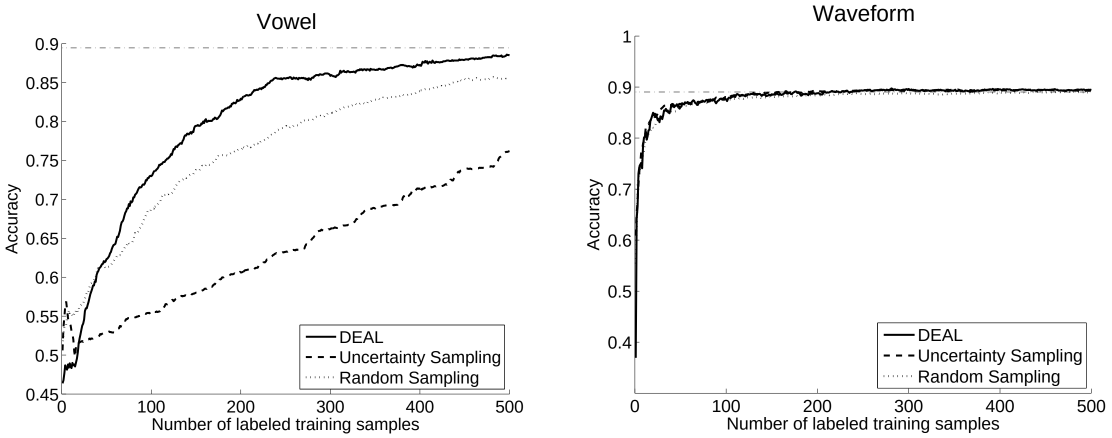


Figure 4.30: Learning curves for data sets “Vowel” and “Waveform”

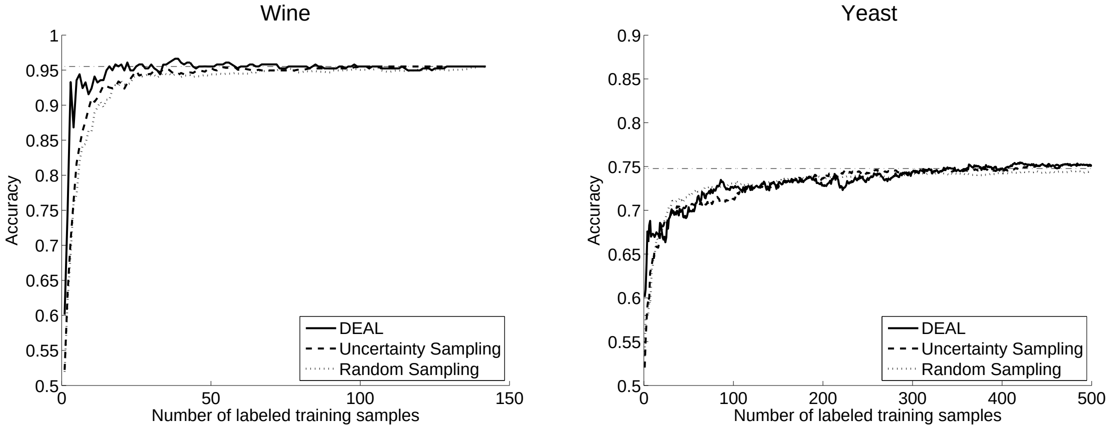


Figure 4.31: Learning curves for data sets “Wine” and “Yeast”

5 Fast Outlier Detection with Random Forests

In this chapter, we present a novel outlier detection algorithm for random forests. To that end, we pick up in idea that has already been introduced in Chapter 3: discriminative learning against additional artificial data from a reference distribution. In contrast to Chapter 3, the method presented here is based on standard RF and can thus be used "out of the box". It is evaluated on toy data and a range of real-world data sets from the UCI machine learning repository [9]. The proposed outlier detection performs significantly better and is faster than Breiman's random forests procedure based on proximity matrices [25].

The method is used later on in Chapter 6 for active learning in industrial quality control.

5.1 Introduction

Real world data often is not as reliable as desired. Among the possible reasons for this are measurement errors, observations not stemming from the intended sample population or miscalculations in data preprocessing. Data analysis may be corrupted by these outliers and thus may lead to wrong conclusions. Hence, identifying outliers is an important first step of statistical analysis. In applications such as fraud detection, outlier detection techniques are even at the core of statistical analysis, as the data is expected to contain unusual samples which are to be detected.

Intrinsically, there can be no universal mathematical definition of "outlyingness": The notion of "outlyingness" is heavily determined by the application and the corresponding interpretation of the data. Moreover, data sets differ in input dimension, variable types, underlying distributions or proportion of outliers. Therefore, a long list of outlier detection schemes have been proposed (see e.g. [32], [16], [83], [85], [77], [177]) and it is advisable to run a "battery of (multivariate) methods" [144], [16] with different properties on a data set to detect anomalies.

In [82], an outlier is described as "an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism". Hence, it may be appropriate to consider as outliers those observations that have a low probability under the distribution estimated from the remaining samples. The suspected outliers can be detected either by standard density estimation methods or, as proposed here, by drawing artificial data from a uniform reference distribution [81, chap. 14] and learning the dichotomy between observed and artificial data, using any classifier which outputs estimates for the posterior class probabilities (see Eq. (5.1)). We show that implementing this idea with random forests maintains the benefits of this classifier, namely the ability of capturing interaction effects [67], [117] and robustness against noise features [110].[1] Moreover, the proposed algorithm has lower computational complexity and performs significantly better on the tested real-world data sets than the original outlier detection scheme proposed by Breiman [25] using the same classifier (Section 5.4.2).

5.2 Random Forests

Consider a classification problem with training set $\mathcal{T} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where the feature vectors $x_1, \ldots, x_n \in \mathcal{X} \subseteq \mathbb{R}^d$ and their class labels $y_1, \ldots, y_n \in \mathcal{Y}$ are independent realizations of the random vector (X, Y).

5.2.1 Algorithm

The random forest classifier [24] is an ensemble learner consisting of M decision trees. To build an individual tree, a bootstrap sample is drawn from the training set and recursively divided until all leaf nodes contain instances of a single class only. For the split at a certain node, $dtry < d$ out of the d feature dimensions are randomly selected and the best split according to the Gini criterion on these $dtry$ variables is used. An estimate for the posterior class probability at an arbitrary point $x \in \mathcal{X}$ is obtained by passing x down all the trees and dividing the number of trees that vote for the respective class by M. The majority vote yields a class assignment. The probability for a training instance of being included in a bootstrap sample is approximately $1 - e^{-1} = 0.632$ (see e.g. [81, chap. 7]). The remaining points are called “out of bag” (oob) for that tree and can be used for an approximate error measurement (oob estimate).

[1] Another advantage is its ease of training; whereas other machine learning algorithms like support vector machines need to be calibrated carefully, random forests are virtually parameter-free: There exists a rule of thumb for the number of split features tried at each node (namely, the square root of the feature space dimension) and the number of trees can be set as high as affordable without the danger of overfitting. (There is some evidence that random forests do overfit [161], but this is certainly not due to too large a number of trees [81, chap. 15].)

5.2.2 Consistency

It is proven in [18] by counter example that random forests (with majority vote) are not consistent (i.e. there exists a distribution of (X, Y) such that the class assignment rule above does not converge to the Bayes classifier (in probability) as n and M tend to infinity).[2] It follows immediately that the estimates for the posterior class probabilities as stated above do not converge to the true posterior class probabilities for every distribution of (X, Y). For $d = 1$, random forests are not even consistent for “non-pathological” examples, e.g. if the class distributions $p(x|Y = y_i)$ are Gaussian. Therefore, we assume that $d \geq 2$ throughout this chapter.

It is emphasized in [17] that it is not known whether random forests are consistent if $d \geq 2$ and if the distribution of X has a Lebesgue density. In either case, it is reasonable and common practice to estimate $p(y|X = x)$ as explained above in Section 5.2.1 and the methods presented in the following sections would not become meaningless if random forests were not consistent. Empirically, random forests output the most accurate estimates for posterior class probabilities among 10 different classification algorithms investigated in [135].

5.2.3 Breiman’s Proposal for Outlier Detection

Let $\mathcal{S}_Z = \{z_1, \ldots, z_{n_Z}\}$ be a given set of d-dimensional observations, some of which may be outliers. An algorithm for detecting these based on random forests has been proposed in [25]. To that end, all the unlabeled observations in $\mathcal{S}_Z$ are assigned to class 1. A second set $\mathcal{S}_B$ of the same cardinality is then created by independently drawing bootstrap samples from the dimension-wise feature values of the first set (thus effectively drawing samples from the product of the empirical marginal distributions). This set is assigned to class 0. A random forest is then trained to discriminate between these two sets, and for each tree it is noted in which leaf node $z_1, \ldots, z_{n_Z}$ end up (no matter if the point is oob or not for a certain tree). The proximity between two points in $\mathcal{S}_Z$ is defined as the number of trees in which they both end up at the same terminal node, divided by two. The proximity between a point and itself is set to 1. An “outlyingness” measure o_i for the points in $\mathcal{S}_Z$ is then obtained by

1. computing o_i as the inverse of the sum of squared proximities between z_i and all points (including z_i) for all $i = 1, \ldots, n_Z$,
2. determining the median of $o_1, \ldots, o_{n_Z}$ and the mean absolute deviation from the median, and
3. normalizing o_i by subtracting the median and dividing by the mean absolute deviation. Values smaller than 0 are set to 0.

[2] It is shown in [26] under which conditions a simplified version of random forests is consistent.

Considering the complexity with respect to the size of $\mathcal{T}$ ($n = |\mathcal{S}_Z| + |\mathcal{S}_B| = 2n_Z$), the actual tree constructions require $O(n \log n)$ operations (see e.g. [81, chap. 9]). But the complexity of the complete algorithm scales as $O(n^2)$ due to the cost of computing the $(n_Z \times n_Z)$-proximity matrix and $o_1, \ldots, o_{n_Z}$.

The algorithm is for example applied in the context of network intrusion detection [189] (in a slightly different manner[3]). An interesting by-product of Breiman's algorithm—not considered further here—is that the proximity matrix defines a Euclidean distance between the observations which can be used for clustering. This has for example been applied successfully in [162].

5.3 Random Forest Outlier Detection

We first briefly recall the idea of transforming density estimation into a supervised learning problem [81, chap. 14] that has already been presented in Section 3.4 and then refine it to obtain a new random forest outlier detection algorithm.

Assume a binary classification problem, i.e. $\mathcal{Y} = \{0, 1\}$. If a classifier outputs an estimate for the posterior class probability $p(y|x)$, Bayes' formula yields

$$p(Y = 1|x) = \frac{p(x|Y = 1)p(Y = 1)}{p(x)} = \frac{p(x|Y = 1)p(Y = 1)}{p(x|Y = 0)p(Y = 0) + p(x|Y = 1)p(Y = 1)}$$

$$\Leftrightarrow p(x|Y = 1) = \frac{p(x|Y = 0)p(Y = 0)}{p(Y = 1)} \cdot \frac{p(Y = 1|x)}{p(Y = 0|x)} \tag{5.1}$$

Let $\mathcal{S}_Z = \{z_1, \ldots, z_{n_Z}\}$ be a set of n_Z realizations of a d-dimensional random vector Z. In order to estimate the density of Z at a point $x \in \mathcal{X}$, we only need to draw a sample $\mathcal{S}_U$ of $n_U = n - n_Z$ points from some known reference distribution whose support encompasses that of the training set distribution, train a classifier on the union set $\mathcal{S}_Z$ vs. $\mathcal{S}_U$ (with labels 1 and 0, respectively) and determine the posterior class probability $p(y|X = x)$. After estimating the class priors $p(Y = 0)$ and $p(Y = 1)$ by n_U/n and n_Z/n, and making a prediction for $p(y|X = x)$, all terms on the right hand side of Eq. (5.1) are approximately known and allow for a density estimate of $Z \equiv [X|(Y = 1)]$ at the point x.

When using random forests for the above scheme, as in 1-nearest neighbor, the tree-wise prediction for a sample is always its label if that sample is not oob for that tree. Hence, when estimating $p(x|Y = 1)$ at a point $x \in \mathcal{S}_Z$, it is advisable to consider only those trees for which $(x, 1)$ is out of bag.

[3] As the data has been labeled with respect to different network services, these labels have been used to train a random forest. The outliers within each class can then be determined without drawing samples from a reference distribution.

As explained in [81, chap. 14], the accuracy of the density estimate depends on the choice of the reference distribution. Here, we simply pick the uniform distribution on a hyper-rectangle $\mathcal{R}$ covering $\mathcal{S}_Z$, i.e. $\mathcal{S}_Z \subset \mathcal{R}$. This may lead to sparse sampling of the artificial data if the feature space dimension d is high and/or the side lengths of $\mathcal{R}$ are large, but it ensures that generating the artificial data is as easy and as fast as for the original method (see Section 5.2.3). Moreover, this simple sampling scheme opens an easy possibility of dealing with both continuous and discrete features in the same data set (see the remark at the end of this section). Note that the number n_U of uniform variates plays the role of a smoothing parameter comparable to the bandwidth of a kernel density estimate: It trades off bias and variance (see e.g. [81, chap. 6]). For random forests, the variance of the estimates can also be decreased by building each tree with a different random sample $\mathcal{S}_U$.

As $p(x|Y=0)$ is constant for a uniform reference distribution, the "degree of outlyingness" can be expressed as a negative monotonic transformation of the density estimates $\hat{p}(x|Y=1)$ for all $x \in \mathcal{S}_Z$. Simply taking the estimates for the posterior class probability $p(Y=0|x)$ is a special case which may make it comfortable to specify a certain threshold above which a point is regarded as an outlier. This also avoids division by 0 in Eq. (5.1).

It follows immediately that, for a "consistent" outlier detection, we do not need to demand that random forests be consistent, but only that $p(Y=1|x) > p(Y=1|x')$ implies

$$\frac{v_{1|x}}{M} > \frac{v_{1|x'}}{M} \text{ as } n, M \to \infty$$

where $v_{i|x}$ denotes the number of tree-wise votes for class i at x (i.e. $v_{0|x} + v_{1|x} = M$).[4]

Computing the density estimate for a point $x \in \mathcal{X}$ costs only $O(n \log n)$ for the tree construction and $O(\log n)$ for the evaluation. Hence, since the number of query points for outlier detection is equal to n_Z, the cost of the proposed method for density estimation is still $O(n_Z \log n_Z)$, whereas Breiman's proposal for outlier detection requires $O(n_Z^2)$ computations (see Section 5.2).

Remark. The derivations in this section have assumed that X and Z are continuous random vectors. If features are discrete (but ordered) or even binary, counting measure and Lebesgue measure are mixed up and sampling the reference class from a rectangle should be avoided. However, this problem is easily solved by drawing random variates from a uniform distribution with respect to counting measure, i.e. from a multinomial distribution with equal probabilities. The feature type can be determined with simple heuristics such that it does not need to be specified manually for a particular data set. All results presented in Section 5.4 are obtained automatically without providing any feature type information.

[4] Note an important difference to the derivations in Chapter 3. There, we counted samples from the training set, whereas here, we count numbers of trees.

5.4 Empirical Evaluation

5.4.1 Data

For motivating the proposed outlier detection scheme, we use various toy data sets. These are described in the following Section 5.4.2. For the actual statistical evaluation, we use 24 different data sets from the UCI machine learning repository [9]. In order to create two-class from multi-class problems, only the two classes with most instances are considered. If classes are equally abundant the ones with the lower class labels are taken. Features that take a single value (i.e. that have zero variance) in the reduced data set are removed. The data is additionally preprocessed as follows: (i) Categorical features with more than two outcomes are replaced by dummy variables, where missing values are treated as a separate outcome, (ii) missing values in continuous features are replaced by the respective mean values and (iii) each variable is normalized to unit variance.

5.4.2 Results

In this section, we compare the proposed outlier detection scheme with the original random forest (see Section 5.2 and [25]) and k-nearest neighbor outlier detection. The latter is chosen for comparison due to easy interpretability and good empirical results [56]. As proposed in [56], we measure the outlyingness of a point $z \in \mathcal{S}_Z$ by the *average* distance to its k nearest neighbors and choose $k = 3$ if not stated otherwise. For the proposed algorithm, the number of trees M is set to 1000 in all experiments (hence, each $z \in \mathcal{S}_Z$ is oob in 368 trees on average) and the hyper-rectangle $\mathcal{R}$ is chosen as the smallest hyper-rectangle that covers $\mathcal{S}_Z + [-0.1, 0.1]^d$. At the beginning, $n_U = n_Z$; the parameter is varied at the end of the section.

5.4.2.1 Fixing $n_U = n_Z$

Toy data We first consider different toy data sets, which favor the proposed method and are chosen to motivate it. Projections of the data sets on their first two dimensions are shown in Fig. 5.1.

In example (a), the "normal" data is uniformly distributed on the 101-dimensional unit hypercube. A single outlier is generated by adding 1 to the first feature of the first of 200 samples. The proposed method identifies the outlier in most of the 200 repetitions (see Table 5.1) as it is relatively far away from the normal samples in the projection to the first dimension. The original outlier detection scheme is only slightly better than pure guessing because the combination of random forests and sampling from the empirical

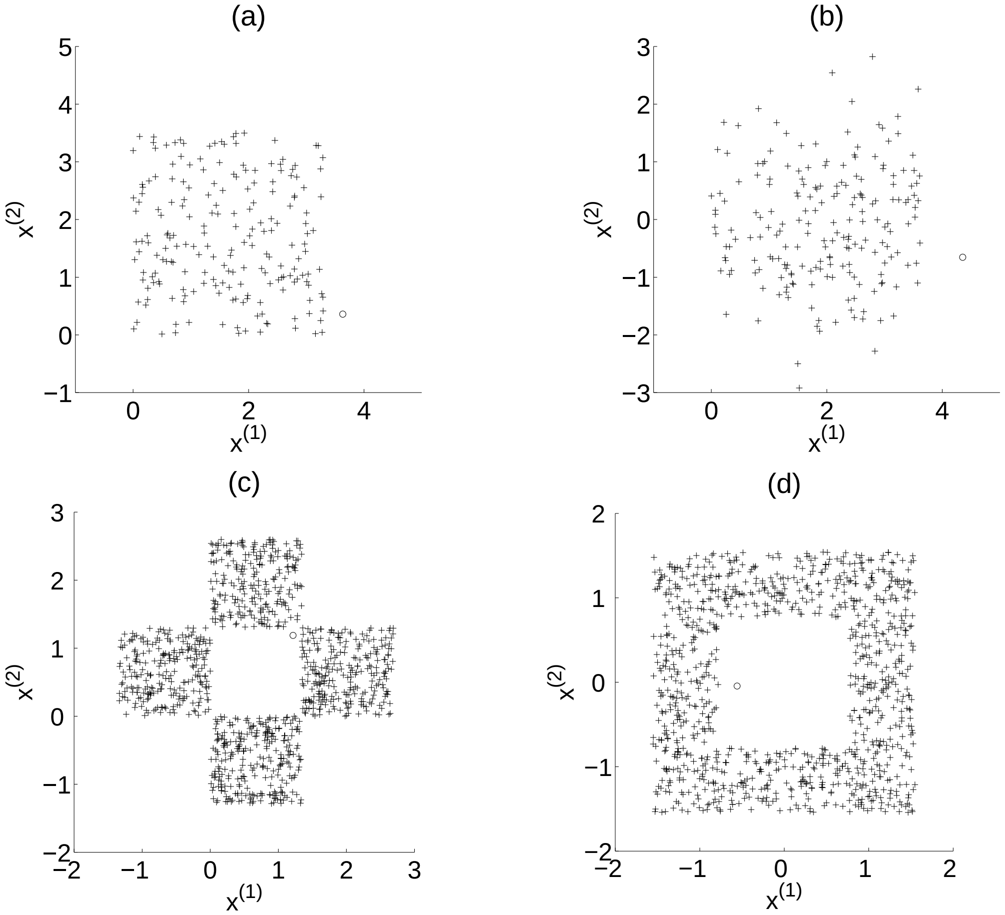


Figure 5.1: The plots show the projections on the first two dimensions of four different toy data examples. "Normal" data (inliers) are represented by crosses, the outlier by a circle. The data generating process is described in the text. Note that the data has been normalized to unit variance in each dimension after sampling.

Example / Method	Median Rank				Mean Rank			
	(a)	(b)	(c)	(d)	(a)	(b)	(c)	(d)
Proposed RF method	1	1	1	1	3.3	12.7	2.6	1.4
Original RF method	94.5	74	1	39	96.1	83.7	10.4	126.2
3-NN	37	15.5	48.5	26.5	49.1	34.6	144.3	130.2

Table 5.1: Median outlyingness rank of the single outlier sample for four different toy data examples. The number of samples n_Z (and thus the range of outlyingness ranks) is equal to 200 in data sets (a) and (b) and equal to 1000 in data set (c) and (d). The number of repetitions is 200.

marginal distributions ignores the distance of the outlier to the normal data in the first dimension; any normal sample that has the lowest or highest value in any dimension is considered as outlying as the outlier itself. The nearest neighbor method suffers from the curse of dimensionality.

Example (b) is a modification of the first one. The first feature is the same is before (including the generation of the outlier) but the other 20 features are independently normally distributed with unit variance. The results are similar but—despite the reduced number of dimensions—the proposed method performs a little bit worse than before since the normal data itself can contain some samples that look like outliers.

In example (c), the first two features of the outlier are drawn from the uniform distribution in the unit square, the “normal” data uniformly from the four adjacent squares. Two independent noise features from the uniform distribution are added. k-nearest neighbor is totally blind to the interaction effect (note in particular the huge difference between mean and median rank; the method completely fails if the outlier is as close to the normal samples as in Fig. 5.1(c)), whereas the median rank of the proposed method is again 1. The original random forest method performs relatively well in this example as many samples of the artificial class fall in the middle square due to sampling from the marginal empirical distributions. This yields a strong discrimination between the two classes in this area.

These things change in example (d), where the first two features of the “normal” data are drawn uniformly from the set $[-2,2]^2\backslash[-1,1]^2$, those of the outlier uniformly from the square $[-1,1]^2$. We add again two independent noise features from the uniform distribution. The original random forest method performs worst now as relatively few samples from the artificial data fall in the square in the middle. The proposed method captures the interaction effect, whereas the nearest neighbor method again fails. Both methods perform better than in the third example because the expected distance between the outlier and the closest “normal” samples is larger than in the example before due to the larger middle square.

Real world data and statistical analysis The statistical analysis on real-world data is based on two-class classification problems. One out of the n_- points of the minority class[5] is added to the n_+ instances of the majority class. After having masked the labels, each of the compared algorithms yields a measure of outlyingness for all $n_+ + 1$ points and all points whose score is above a certain threshold are considered an outlier. This is repeated n_- times for all instances of the minority class and ROC curves are obtained by varying the threshold.

The data set “Balance-Scale” [167] (available from the UCI machine learning repository [9]) has been generated with strong interaction effects to model psychological

[5] In case of class balance, the class with smaller label value is chosen as “outlier class”.

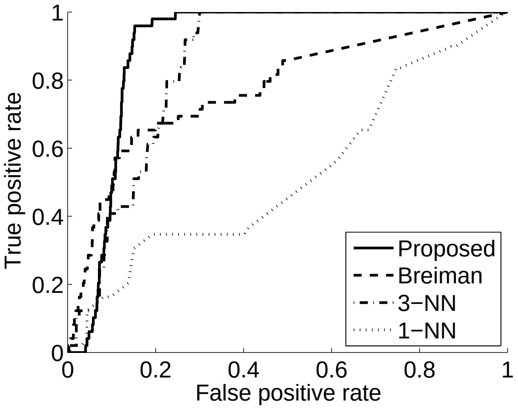


Figure 5.2: ROC curves of the data set Balance-Scale when using classes “L” and “B”

experiments. Each of the 4 features can take the values $1, \ldots, 5$ and a sample belongs to class “L” (“Left”, 288 samples altogether) if the product of the first to two features is larger than the product of features 3 and 4. A sample belongs to class “B” (“Balance”, 49 samples) if the products are equal.[6] The ROC curves for this data set are shown in Fig. 5.2. It can be observed that the proposed algorithm captures the interaction effect best. The 1-nearest neighbor outlier detection algorithm performs as bad as pure guessing. Areas under curve are 0.896, 0.775, 0.847 and 0.523 in the order of the labels in the legend.

The areas under curve (AUC) for 24 UCI data sets are presented in Table 5.2 and Fig. 5.3. Note that some of the ostensibly poor results are due to significant overlap of the respective classes in feature space. In order to compare the results statistically, we employ a simple two-sided sign test [45]. The proposed method performs significantly better than Breiman’s method for outlier detection ($p = 2.77 \cdot 10^{-4}$), even if the algorithm (which first computes a proximity matrix and then determines the outliers based on these distances) is combined with the proposed uniform sampling scheme[7] ($p = 0.023$). Recall from Section 5.3 that the algorithm has lower computational complexity, too. The results of the 3-nearest neighbor method do not differ significantly from the proposed detection scheme ($p = 0.678$).

[6] The 288 samples of the remaining class “R” have been discarded for this analysis here to make the task more difficult. The results for the standard procedure of using the two most abundant classes (see Section 5.4.1) —here “L” and “R”—are reported in Table 5.2.

[7] It is mentioned in [166] that a uniform sampling scheme was included in an earlier version of Breiman’s FORTRAN code and that it is still implemented in the R package “randomForest” [107] as a second option.

Dataset	AUC$_{\text{prop}}$	AUC$_{\text{Br}}$	AUC$_{\text{Br,unif}}$	AUC$_{\text{3-NN}}$	n_+/n_-	d
Anneal	0.753	0.965 (–)	0.473 (+)	0.963 (–)	684/99	57
Audiology	0.966	0.757 (+)	0.923 (+)	0.956 (+)	57/48	66
Autos	0.792	0.537 (+)	0.711 (+)	0.770 (+)	67/54	63
Balance-Scale	0.994	0.959 (+)	0.983 (+)	0.989 (+)	288/288	4
Breast-W	0.979	0.793 (+)	0.985 (–)	0.991 (–)	458/241	9
Dermatology	1.000	0.928 (+)	1.000 (+)	1.000 (±)	112/72	33
Ecoli	0.979	0.569 (+)	0.957 (+)	0.989 (–)	143/77	6
Heart-C	0.832	0.682 (+)	0.842 (–)	0.786 (+)	165/138	23
Hepatitis	0.800	0.699 (+)	0.846 (–)	0.819 (–)	123/32	39
Ionosphere	0.966	0.953 (+)	0.950 (+)	0.974 (–)	225/126	33
Iris	1.000	0.847 (+)	0.983 (+)	1.000 (±)	50/50	4
Led24	0.935	0.838 (+)	0.934 (+)	0.929 (+)	114/110	24
Letters	0.993	0.969 (+)	0.956 (+)	0.998 (–)	813/805	16
Lymph	0.802	0.772 (+)	0.801 (+)	0.777 (+)	81/61	37
Optdigits	0.950	0.987 (–)	0.934 (+)	0.984 (–)	572/571	56
Pendigits	1.000	0.999 (+)	1.000 (+)	1.000 (±)	780/780	16
Satimage	0.983	0.864 (+)	0.996 (–)	0.999 (–)	1533/1508	36
Segment	0.953	0.934 (+)	0.920 (+)	0.979 (–)	330/330	18
Soybean	0.884	0.854 (+)	0.795 (+)	0.886 (–)	92/91	56
Vehicle	0.943	0.894 (+)	0.847 (+)	0.949 (–)	218/217	18
Vote	0.845	0.421 (+)	0.902 (–)	0.782 (+)	267/168	48
Vowel	0.984	0.992 (–)	0.573 (+)	0.997 (–)	90/90	27
Waveform	0.810	0.604 (+)	0.820 (–)	0.798 (+)	1692/1655	40
Wine	0.954	0.597 (+)	0.943 (+)	0.943 (+)	71/59	13

Table 5.2: Comparison of the areas under ROC curve of 24 different real-world data sets between—from left to right—the proposed $O(n \log n)$ method, Breiman's $O(n^2)$ outlier detection, Breiman's $O(n^2)$ outlier detection combined with the proposed uniform sampling scheme and 3-nearest neighbor outlier detection. For each data set, the algorithms are run n_- times on a set with $n_Z = n_+ + 1$ observations and results are averaged over all these runs. (+) and (–) indicate whether the proposed method performs better or worse, respectively ((±) in case of equality). Compared over all data sets, the proposed method performs significantly better than both variants of Breiman's algorithm and not significantly differently from the nearest neighbor method. Some of the results are also presented graphically in Fig. 5.3.

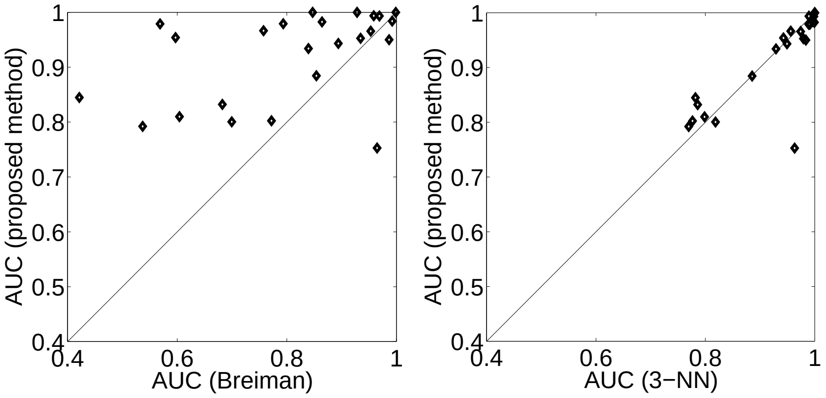


Figure 5.3: Graphical comparison between the results of the proposed method, Breiman's outlier detection scheme and a nearest neighbor method. Each diamond represents one data set. It can observed that the two methods based on random forests perform quite differently, whereas the results of the proposed method are similar to the ones of 3-nearest neighbor.

5.4.2.2 Varying n_U

Now, we investigate the influence of n_U on the performance of the proposed algorithm. To that end, we vary the "sample factor" $s = n_U/n_Z$ and run the same simulations as above with $s = 0.1, 0.3, 1, 3, 10$. The results are reported in Table 5.3 and it can be observed that they are not very sensitive with respect to the number of additional points from the reference class. The data sets "Anneal" and "Waveform" are notable exceptions, where the AUC differences between best and worst results are larger than 0.1. Interestingly, although the feature space dimension d is relatively high for both data sets and the number of observations n_Z is in the same range, the "Anneal" results improve for increasing s, whereas the "Waveform" results deteriorate. In Fig. 5.4, it is investigated if there is a connection between the feature space dimension d and the optimal sample factor s.

Despite the robustness of the method with respect to the parameter choice, it may be reasonable to optimize s, for example by cross-validation methods. The most obvious possibility is to use the oob error for the optimization. But this is a bad idea as it favors extreme settings: The classification error is very low if s is very small or very large and almost all oob samples are classified as 1 or 0, respectively. Instead, one can use the area under the ROC curve as optimization criterion, which is invariant to a priori class probabilities [23].[8] Note that, if the number of samples from the artificial class is low, additional samples from this class can be generated for the estimation of the

[8] Note that the AUC used here for parameter optimization and the AUC used for performance comparison of the outlier detection methods are different. Here, the computation of the AUC is based on a "standard" supervised two-class learning setting.

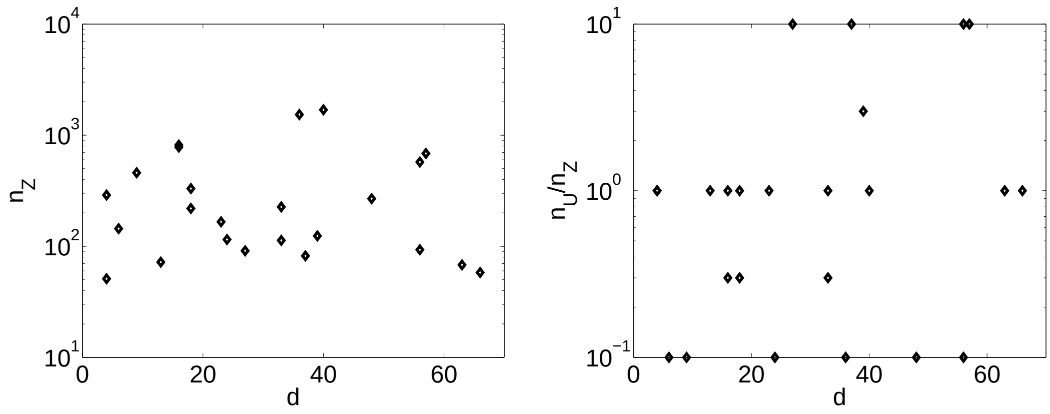


Figure 5.4: The left panel compares the number of observations n_Z of a data set with its feature space dimension d. Each diamond represents one data set. No particular connection can be observed. This is important for interpreting the plot in the right panel, where the connection between the optimal "sample factor" $s = n_U/n_Z$ (see results in Table 5.3) and d is investigated. There seems be a small positive correlation between these two, but there is certainly no strong dependence.

AUC; these are—of course—not used for determining the outliers. Unfortunately, this cross-validation scheme does not yield satisfactory results, as reported in Table 5.3.

However, $s = 1$ is a good default choice with "Anneal" being one exception. For this data set, the proposed (uniform) sampling scheme seems to yield too few artificial data in the relevant regions in feature space. This can be inferred from Table 5.2 and Fig. 5.3: Breiman's method achieves much better results for this data set when combined with his sampling scheme.

5.5 Conclusions

In this chapter, we have presented an improved outlier detection scheme for random forests. The method is based on the idea of adding random variates from a reference distribution—which is uniform in our case—to the initial dataset.

Using various toy data sets, we have shown that interesting properties of random forests, namely the ability of capturing interaction effects and robustness against feature noise, carry over to the proposed outlier detection scheme. A comparison to other methods has been carried out on 24 real-world data sets from the UCI machine learning repository. The outlier detection performance of the proposed method is similar to a standard nearest neighbor novelty detection scheme and it performs significantly better

Dataset	$s = 1$	maxΔ	s	minΔ	s	CV	s	d
Anneal	0.753	0.142	10	-0.076	0.1	0.894 (+)	10	57
Audiology	0.966	0.000	1	-0.006	0.1	0.966 (–)	10	66
Autos	0.792	0.000	1	-0.055	0.1	0.781 (–)	3	63
Balance-Scale	0.994	0.000	1	-0.007	0.1	0.987 (–)	0.1	4
Breast-W	0.979	0.009	0.1	-0.001	3	0.988 (+)	0.1	9
Dermatology	1.000	0.000	1	0.000	0.1	1.000 (±)	10	33
Ecoli	0.979	0.011	0.1	-0.038	10	0.953 (–)	3	6
Heart-C	0.832	0.000	1	-0.023	0.1	0.829 (–)	3	23
Hepatitis	0.800	0.036	3	-0.016	0.1	0.800 (±)	1	39
Ionosphere	0.966	0.011	0.3	-0.040	10	0.966 (±)	1	33
Iris	1.000	0.000	1	-0.034	10	1.000 (±)	1	4
Led24	0.935	0.022	0.1	-0.023	10	0.957 (+)	0.1	24
Letters	0.993	0.001	0.3	-0.002	0.1	0.992 (–)	10	16
Lymph	0.802	0.007	10	-0.021	0.1	0.809 (+)	10	37
Optdigits	0.950	0.013	0.1	-0.017	10	0.933 (–)	10	56
Pendigits	1.000	0.000	1	-0.000	10	1.000 (±)	3	16
Satimage	0.983	0.016	0.1	-0.013	10	0.969 (–)	10	36
Segment	0.953	0.000	1	-0.022	10	0.931 (–)	10	18
Soybean	0.884	0.014	10	-0.033	0.1	0.884 (±)	1	56
Vehicle	0.943	0.021	0.3	-0.024	10	0.965 (+)	0.3	18
Vote	0.845	0.028	0.1	-0.023	10	0.872 (+)	0.1	48
Vowel	0.984	0.011	10	-0.070	0.1	0.995 (+)	10	27
Waveform	0.810	0.000	1	-0.114	10	0.753 (–)	3	40
Wine	0.954	0.000	1	-0.028	0.1	0.954 (±)	1	13

Table 5.3: The results of the proposed outlier detection method for varying "sample factor" $s = n_U/n_Z$. $s = 1$ corresponds to the parameter choice in the previous subsection. The column "maxΔ" ("minΔ") shows the difference between the best (the worst) result among all s and the default choice $s = 1$; the corresponding value for s is reported to the right of the column. The column "CV" shows the AUC if s is determined according to a (unsuccessful) cross validation scheme. "+", "–" and "±" indicate if there is an improvement over the default choice $s = 1$. The last column shows the dimension of the data.

than Breiman’s method based on a random forests proximity measure. In addition, the proposed scheme has lower computational cost than Breiman’s method.

In the next chapter, the proposed outlier detection algorithm is used as a part of a new active learning strategy for defect detection.

6 An Active Learning Strategy for Defect Detection

Human industrial quality control is always subjective and sometimes error-prone, and the costly procedure may be automated using algorithmic analysis of images of production parts. If supervised classification algorithms are employed, many examples of defective and intact parts need to be provided for classifier training. This calls for the application of active learning. In this chapter, we present a novel active learning strategy that addresses three challenges in defect detection: initial absence of labels, class imbalance and weak labels. It is implemented with standard random forest after extending the training set with additional samples from a reference distribution as already made use of in Chapters 3 and 5. The method achieves steep learning curves on the DAGM contest benchmark data set.

6.1 Introduction

Quality control is an integral part of industrial mass production. To prevent the sales of defective products, every single part must be inspected. Typical tasks in industrial quality control include completeness checks, precision measurements or surface inspection. In order to automate the time-consuming, costly and subjective procedure of *human* inspection [158], an image can be taken of each part and subjected to algorithmic analysis. On the long list of possible applications are the assessment of steel [145], stone countertops [113], fabric [99], wood [168], ceramic tiles [186], cork [61], diode chips [109] or semiconductors [103]. Here, we specifically concentrate on defects such as scratches, stains and other irregularities on surfaces with stochastic texture.

Various methods for automated defect detection on textured surfaces have been proposed; see [185] for a comprehensive review. Similar to the methods presented in [3], [94], [97], [118], [146] and [180], we employ a learning-based approach. As will be explained in detail in Section 6.4, an image is divided into several patches, and each patch is represented by a point in feature space.

For training, a statistical classifier for automated inspection requires a set of sample patches (and thus images) of defective and intact parts, along with labels. We propose to minimize the labeling effort in two ways: by requiring only “weak” labels that a human expert can specify with little effort; and by obviating the labeling of parts with little novelty for the classifier.

More specifically, the definition of “weak” labels used here is as follows: the human annotator is instructed to provide labels not at pixel-precision, but by generously outlining a defect in the image with an ellipse (see Fig. 6.1). This approximate labeling leads to false positive pixels, i.e. the ellipse may contain both defective and intact pixels, whereas all pixels outside an ellipse should be intact.

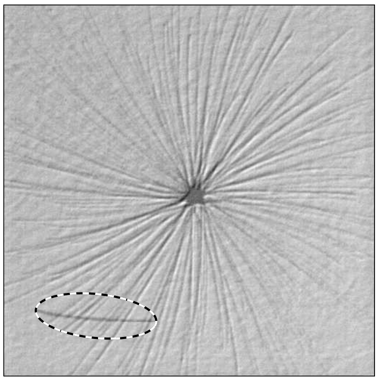


Figure 6.1: Example of a weakly labeled image

To further reduce the labeling effort, we recur to active learning. An AL strategy that is applied in the field of industrial optical inspection needs to comply with several requirements. In the first iterations of standard AL algorithms, the samples to be labeled are often selected at random until at least one labeled example of each class is available. However, a distinctive feature of industrial inspection is that, by economic necessity, most samples from a production line are intact. Accordingly, if the imbalance between the negative/intact and the positive/defect classes is extreme, chances of capturing a positive example in this random set are slim. Therefore, an appropriate AL strategy should somehow explore the feature space from the beginning, i.e. before having obtained a label from each class.

Further, if the penalty associated with false negative predictions by the trained classifier is high (as it is for any quality-conscious manufacturer), it is natural for the AL strategy to employ a biased notion of “informativeness” and request labels particularly for those samples that will help refine the decision boundary in a way that prevents false negative predictions. To this end, the strategy should request labels in the vicinity of previously found positive examples where the decision boundary is not yet well determined (“exploitation”); but it should also request labels in those areas of feature space that are far from the known labels of either class, to avoid overlooking positive examples (“exploration”).

Finally, one feature of the setting described here is that labels always come in bags: the proposed classifier acts on image patches, but the oracle labels entire images.

In summary, the AL strategy presented here is designed for

- active learning with
- weak labels that
- come in bags and reflect
- strongly imbalanced classes.

To the best of our knowledge, the proposed AL approach is the first that responds to the demands in defect detection for industrial quality control. Although the AL strategy is tailored to this application, it is of course not restricted to the domain of defect detection and may be applied to similar classification problems with imbalanced classes, bag-wise and/or unreliable labels.

In Section 6.2, the generic AL strategy is presented in detail and a specific proposal for its instantiation is made in Section 6.3 based on the random forest classifier [24]. In Section 6.4, the representation of images in feature space and other technical details that are relevant for the application of the approach in the area of defect detection are elaborated. Experimental results are presented in Section 6.5.

6.2 A Novel Active Learning Strategy for Defect Detection

6.2.1 Problem Setup and Notation

Let (X,Y) be a random vector with distribution $p(x,y)$, where $x \in \mathcal{X} \subseteq \mathbb{R}^d$ is a feature vector and $y \in \{-1,1\}$ its true class label. Let $z \in \{-1,1\}$ be the (possibly wrong) weak label of a point or patch x, obtained from the human annotator. Points with label $z=-1$ and $z=1$ will be referred to as negative and positive, respectively. It follows from the instructions to the human annotator described in the introduction that negative labels are reliable whereas positive ones are not.

Further, let a bag/image B contain m points/patches, i.e. m realizations of (X,Y). The bag B is called positive if it contains at least one point with true label 1, i.e. if $|\{(x,y) \in B : y=1\}| \geq 1$, negative otherwise.[1] A positive bag corresponds to a defect image, a negative bag to an image of an intact part.[2]

[1] Note that, since wrong labels only occur in positive bags, the labels z (instead of y) could equivalently be used to define if a bag is positive or not.

[2] Note that this setting is very related to "multiple-instance learning" [5, 50], where some terms introduced here are taken from. The only difference is that, in a standard setting of multiple-instance learning, there is less information about the positive bags: only the image itself would be labeled, corresponding to an "ellipse" which comprises the whole image.

We assume that there is a large pool $\mathcal{U}$ of unlabeled bags available at the beginning of the AL process and that all bags—labeled and unlabeled—are known prior to AL.[3] Moreover, there is a small (possibly empty) set $\mathcal{L}$ of labeled points, selected from previously labeled bags. Note that $\mathcal{U}$ contains bags, whereas $\mathcal{L}$ contains points.

Three significant challenges are pertinent to the application of AL in defect detection:

1. The (initially unlabeled) training data is highly imbalanced. There are many more negative than positive bags in $\mathcal{U}$. Moreover, while the intact images (and the intact regions of defect images) share some common stochastic texture, the defects may look very different. Thus, the positive points are not only rare but may also be widely spread in feature space.

2. Some labels are false positive due to the weak labeling introduced in Section 6.1. In contrast, as discussed above, the labels of the negative samples are reliable.

3. Labels can only be provided in bags of m data points since only entire images are labeled on request.

The AL strategy presented below responds to these challenges. To avoid a potpourri of heuristics, we assume that candidate methods that are used for the instantiation of the strategy have the following two capabilities:

(C1) During the AL process (particularly at the beginning), there may only be negative points in $\mathcal{L}$. Hence, for each unlabeled point x, the method should return some measure $\hat{o}(x)$ of "outlyingness" that evaluates how much the feature vector x is consistent with the samples in the current training set $\mathcal{L}$. We assume that large values of $\hat{o}(\cdot)$ correspond to a high degree of "outlyingness". Expressed in a different way, the method should be able to perform one-class learning with the additional ability of stating some confidence regarding the decision whether x belongs to the learned class or not.

(C2) At some point of the AL process, there are labeled examples of both classes available. We demand that, for each x, the method returns an estimate $\hat{p}(Y = 1|x)$ for the posterior class probability $p(Y = 1|x)$. In addition, the method should quantify its uncertainty about this estimate, i.e., some measure $\hat{u}(x)$ should be returned that is related to the number of samples from the current training set $\mathcal{L}$ in the neighborhood[4] of x. We assume that large values of $\hat{u}(\cdot)$ correspond to high uncertainty or, expressed in a different way, low confidence.

[3] As explained in Section 2.2, this scenario is called *pool-based* AL.

[4] As discussed in Chapter 3, the notion of neighborhood depends on the employed learning algorithm.

6.2.2 Proposed Active Learning Strategy

Our response to the challenges mentioned in Section 6.2.1 revolves around a two-step AL strategy. In the first step (referred to as "query step" in the following), the bag B to be labeled is selected from $\mathcal{U}$. After having obtained the labels of the points in B from the human annotator, a second step ("elimination step") follows during which only a subset $B_{\mathcal{L}} \subset B$ of the requested labeled data points is actually added to the training set $\mathcal{L}$. This subset is selected via a criterion that retains only those data points that are most important for learning the decision boundary and the subsequent iterations of the AL process. At the same time, the criterion helps the learning algorithm to steer clear from overfitting to false positive labels by discarding them from the training set $\mathcal{L}$.[5] As a side effect of this selection process, the size of $\mathcal{L}$ and thus computation time for updating the classifier is considerably reduced. An overview of the complete AL procedure is given in Fig. 6.2. It will be explained in detail in the rest of the section.

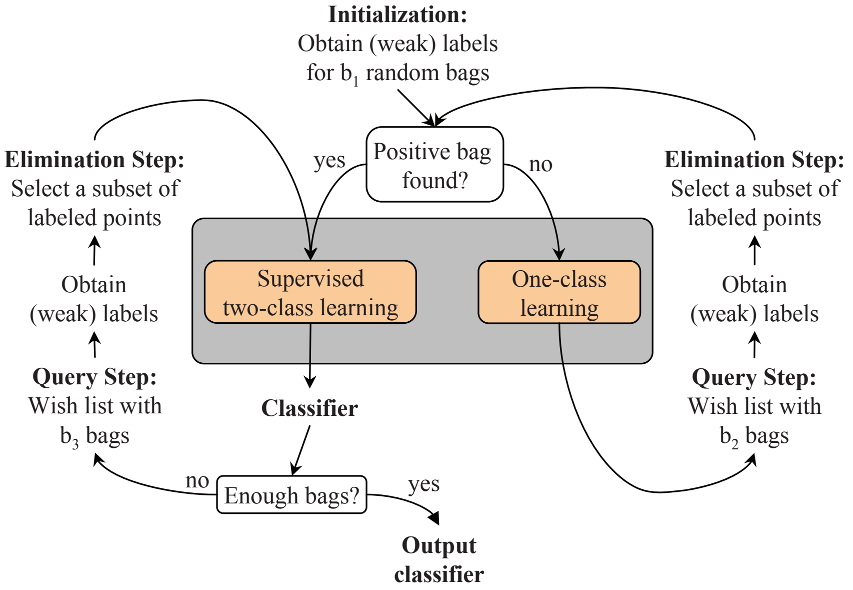


Figure 6.2: Overview over the complete AL process. Details are given in the text.

6.2.2.1 Initialization

In practice, $\mathcal{L}$ is usually *not* empty prior to the AL process. For example, a few images of production parts may have been taken to test the whole machinery for image acquisition or the labeling tool. Some or all of the corresponding training points can be added to $\mathcal{L}$.

[5] It has been shown empirically that classification performance can be improved if wrong labels are eliminated from the training set (see e.g. [28, 195, 179]).

The AL process is already initialized at this point and we can proceed with the query step as described below.

If $\mathcal{L}$ is empty for some reason, we obviously need to decide on the first one or—more generally—the first b_1 bags from $\mathcal{U}$ that labels are requested for. Unsupervised learning techniques like outlier detection [83, 13] or cluster analysis [57, 65] could be used to find interesting structures in the unlabeled data and thus to choose those b_1 bags that minimize or maximize some optimality criterion. For sake of simplicity, the AL process may be initialized by drawing b_1 bags from $\mathcal{U}$ at random.

There are two possible outcomes: either all b_1 bags are negative or at least one of them is positive. As illustrated in Fig. 6.2, if all labeled bags turn out to be negative, i.e. the $b_1 \cdot m$ initial data points are realizations of the distribution of the negative class, the algorithm proceeds with one-class learning (C1). If there are samples of the positive class, the algorithm proceeds with supervised learning (C2). In either case, we proceed with a query step as described in the following.

6.2.2.2 Query Step

In the query step, bags of unlabeled examples are selected for subsequent labeling. It is known a priori that most of the unlabeled bags in $\mathcal{U}$ are negative. In contrast, positive bags are very rare and thus (almost) all of them are important for learning an appropriate classifier. Therefore, the priority of the query step is to find as many positive bags and thus positive samples as possible.

As discussed above, we need to distinguish two cases: either (i) all points in $\mathcal{L}$ are negative or (ii) there is at least one positive point in $\mathcal{L}$.

In the first case, we apply one-class learning. The basic tenet is that, if the positive and negative class distributions do not overlap too much, outliers are more likely to have a positive label than others. So, unlabeled data points that are least consistent with the points in $\mathcal{L}$ are the most interesting ones. According to (C1), the employed learning algorithm indeed has the capability of estimating the "outlyingness" $\hat{o}(x)$ of each unlabeled point x. This property is made use of in the following.

In general, not much may be known a priori about the characteristics of the defects that can occur. If a defect is small, it comprises only a few, possibly only one of the patches of the defect image and $\hat{o}(x)$ is relatively *small* for almost all of the data points in a positive bag. Hence, it is sensible to consider only the most "extreme" data point in each bag and to define the training utility of a bag B as

$$TUV(B) := \max_{x \in B} TUV(x) = \max_{x \in B} \hat{o}(x) \tag{6.1}$$

Then, for subsequent labeling, the b_2 bags in $\mathcal{U}$ with the largest training utility are chosen.

Once both positive and negative bags are available (ii), the algorithm proceeds with supervised two-class learning (Fig. 6.2). On one hand, we want to obtain more positive labels in the neighborhood of those that have already been found in order to refine the decision boundary ("exploitation"). The degree of vicinity of a point x to the positive labels in $\mathcal{L}$ can be measured by the estimate $\hat{p}(Y = 1|x)$ for the posterior class probability of the positive class. On the other hand, positive points may be highly spread in feature space and exploring the feature space thoroughly is crucial for discovering other regions of positive labels. Therefore, we want to obtain labels for data points which are far away from the labeled points of either class in feature space ("exploration"). According to (C2), the employed learning algorithm returns the corresponding measure $\hat{u}(x)$ in addition to $\hat{p}(Y = 1|x)$.

According to the considerations above for case (i), only a small minority of the points in a positive bag may be positive. Hence, we again consider only the most "extreme" data point in each bag and define the training utility of a bag B as

$$TUV(B) := \max_{x \in B} TUV(x) = \max_{x \in B} \left[\hat{p}(Y = 1|x) + \gamma \hat{u}(x)\right], \quad \gamma > 0 \tag{6.2}$$

Then, for subsequent labeling, the b_3 bags in $\mathcal{U}$ with the largest training utility are chosen. The user-defined parameter γ trades off exploitation and exploration. It reflects the prior belief in the similarity of possible defects. The more dissimilar different defects are, the larger γ should be chosen.

The training utility function in Eq. (6.2) is a heuristic. Another possibility would have been to use a function of the product of prediction and exploration term as in [129]. A very similar training utility function to the one used here has for example been proposed in [31].

The parameters b_2 and b_3, which define the number of bags presented to the human annotator after each iteration of the selection process, are user-defined. The parameters trade off efficiency of the AL process, computation time and the availability of the human annotator. Larger values lead to less efficiency (the label information of the b_2 or b_3 bags in the same batch may be "overlapping"), shorter computation time (fewer wish lists need to be generated for the same number of bags) and more convenient labeling (it is more acceptable to provide many labels in one go than to continually prompt the labeler at a certain interval).

6.2.2.3 Elimination Step

After having been provided with (weak) labels for all b_2 or b_3 bags by the human annotator, the algorithm proceeds with the elimination step of the AL process. During this step, only a subset $B_{\mathcal{L}}$ of the labeled data points of each bag B is actually added to the training set $\mathcal{L}$. In doing so, we pursue four goals.

(G1) $B_{\mathcal{L}}$ should contain the correct positive labels, i.e. the points $\{(x, y, z) \in B : y = 1, z = 1\}$.

(G2) $B_{\mathcal{L}}$ should not contain any wrong positive label, i.e. any of the points $\{(x, y, z) \in B : y = -1, z = 1\}$.

Among the negative points, $B_{\mathcal{L}}$ should contain at least those that are either

(G3) distant to the points in $\mathcal{L}$ to memorize that a certain region in feature space region has already been explored or

(G4) close to the decision boundary to optimize classification performance.

We first concentrate on the goals (G1) and (G2). Consider a labeled positive bag B. For each point $(x, y, z) \in B$, we have estimated either $\hat{o}(x)$ (Eq. (6.1)) or $\hat{p}(Y = 1|x) + \gamma\hat{u}(x)$ (Eq. (6.2)) in the query step. These two estimates can be exploited again to perform a statistical test. First, recall that the negative labels are reliable. If the patches of an image do not overlap too much and if the texture is sufficiently repetitive, the negative points in a bag can be regarded as independent realizations of the negative class conditional $X|Y = -1$. Then the distribution of $TUV(X|Y = -1)$ can easily be estimated by its empirical distribution $\hat{F}$. Hence, the $(1 - \alpha)$-quantile of $\hat{F}$ is an approximate critical value c_α for a level-α-test on the hypothesis that x has been drawn from the negative class conditional $X|Y = -1$. If $TUV(x) > c_\alpha$, the hypothesis can be rejected. This means that the true label y of the positive points $(x, y, z) \in B$ for which $TUV(x) > c_\alpha$ is 1 with high probability. These points are added to $\mathcal{L}$ (G1). If $TUV(x) \leq c_\alpha$ for a positive point, the hypothesis cannot be rejected and nothing can be implied. However, if the test statistic $TUV(x)$ discriminates well between the positive and the negative class, the power of the test is large and $y = -1$ for most of the positive points for which $TUV(x) \leq c_\alpha$. These points are *not* added to $\mathcal{L}$ (G2).

We note for later simplification that, if the number of positive points is small compared to the total number m of points in the bag, we can simply add those positive points x to $\mathcal{L}$ for which $TUV(x)$ is among the $(1 - \alpha) \cdot 100\%$ largest values of all points in B, i.e. c_α is computed using both positive and negative points.

Next, consider a labeled bag B, positive or negative, to discuss the goals (G3) and (G4). By the definition of a level-α-test, the hypothesis of having been drawn from negative class conditional $X|Y = -1$ is rejected for $\alpha \cdot 100\%$ of the negative points. However, we know that the true label of these points definitely is negative. This makes these points highly interesting with respect to the goals (G3) and (G4): If, on one hand, $\hat{o}(x)$ or $\hat{u}(x)$ is large, then x is far away from data points of either class in the training set $\mathcal{L}$ (G3). If, on the other hand, $\hat{p}(Y = 1|x)$ is large, then x is relatively close to the current decision boundary (G4). Hence, we add those negative points in B to $\mathcal{L}$ for which $TUV(x)$ is large.

In summary, to pursue the goals (G1)-(G4), we simply add those $\alpha \cdot 100\%$ of the points in B for which $TUV(x)$ is largest, i.e.

$$B_{\mathcal{L}} = \{(x, y, z) \in B : TUV(x) > c_\alpha\}$$

This is illustrated in Fig. 6.3. In statistics, $\alpha = 0.05$ is commonly used as significance level. We propose to use the same value here.

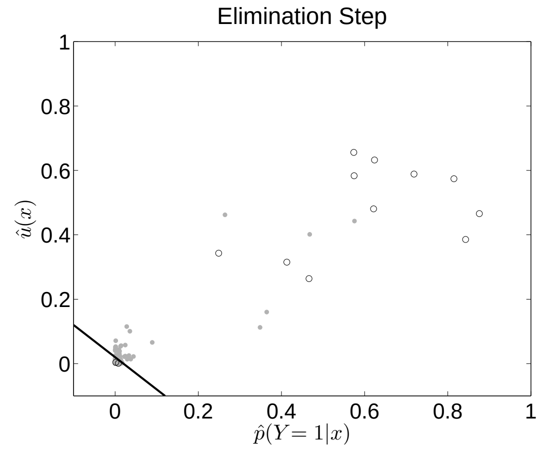


Figure 6.3: Elimination step of the AL strategy. The points $x_1, \ldots, x_m$ of a just labeled bag B are depicted as circles; negative points are filled and gray, positive ones are open and black. They are represented in the two-dimensional space spanned by the posterior prediction $\hat{p}(Y = 1|x)$ and the measure $\hat{u}(x)$ for the uncertainty about the prediction. The points above the black line $\hat{p}(Y = 1|x) + \hat{u}(x) = c_{0.05} = 0.02$ (γ is set to 1 here) are added to $B_{\mathcal{L}}$. The points below the line are not added to $\mathcal{L}$, among them two positive points which are incorrectly labeled with high probability. Note that—in contrast to the visual impression—95% of the points are below the threshold line.

6.3 Instantiation of the Active Learning Strategy using Random Forests

To implement the AL strategy presented in the previous section, we need a method that (C1) returns a measure $\hat{o}(x)$ for the “outlyingness” of a test sample x if the training set $\mathcal{L}$ contains negative samples only, and that (C2) returns an estimate $\hat{p}(y|x)$ for the posterior class probabilities at a test sample x together with a confidence $\hat{u}(x)$ in this estimate if $\mathcal{L}$ contains both positive and negative samples. In this section, we discuss how a random forest classifier [24] (RF) can be extended such that it has these capabilities. For the chapter to be self-contained, we first briefly introduce standard RF in Section 6.3.1. Afterward, we present modified versions for one-class (Section 6.3.2) and two-class learning (Section 6.3.3).

6.3.1 Random Forests

A random forest is a learning algorithm that consists of an ensemble of M decision trees $\{h_i\}_{i=1}^{M}$. To build an individual tree, a bootstrap sample is drawn from the training set $\mathcal{L}$ and defined as the root node. At each node of the tree, $dtry$ out of the d features are randomly chosen and the best axis-orthogonal split according to the Gini criterion among these $dtry$ variables is used to divide the set of training samples at this node into two parts. These two subsets then constitute the two children of the node. In contrast to many other tree-based learning algorithms, the procedure is continued until all leaf nodes are pure, i.e. all training samples in a leaf node have the same class label. To obtain a vote $h_i(x)$ from an individual tree, a test sample x is passed down the tree to its leaf node and the unique label of the training samples in this node is assigned. The average number of trees that vote for a certain class is usually used as an estimate for the posterior class probability of this class, i.e.

$$\hat{p}(Y = y|x) = \frac{1}{M} \sum_{i=1}^{M} \mathbf{1}\{h_i(x) = y\}$$

where $\mathbf{1}\{\cdot\}$ denotes the indicator function. Given equal misclassification costs, a crisp class assignment is obtained from the majority vote:

$$h(x) = \arg\max_y \sum_{i=1}^{M} \mathbf{1}\{h_i(x) = y\} = \arg\max_y \hat{p}(Y = y|x)$$

A main advantage of random forests is their ease of training; whereas other machine learning algorithms such as support vector machines need to be calibrated carefully, random forests are virtually parameter-free: There exists a rule of thumb for setting $dtry$ (namely, $dtry = \sqrt{d}$) and the number M of trees can be chosen as high as affordable without the danger of overfitting.[6] Moreover, random forests are not restricted to binary problems, but can handle multi-class settings equally well.

6.3.2 One-Class Learning

If the set of currently labeled samples $\mathcal{L}$ contains negative points only, we want to identify those unlabeled points/bags that are least consistent with the samples in $\mathcal{L}$ for subsequent labeling (C1). To this end, we apply the outlier detection method presented in Chapter 5.

Let $\mathcal{L} = \{(x_i, -1)\}_{i=1}^{n}$ be the current training set and let $\mathcal{R}$ be a rectangle that covers the training set and all unlabeled points, i.e. $\{x_i\}_{i=1}^{n} \cup \{x \in B : B \in \mathcal{U}\} \subset \mathcal{R}$. Further,

[6] There is some evidence that random forests do overfit [161], but this is certainly not due to too large a number of trees [81, chap. 15].

let $\mathcal{S}_0 = \{x_i\}_{i=n+1}^{n+n_0}$ be a set of n_0 realizations from the uniform distribution on $\mathcal{R}$. These samples are labeled as 0 and we obtain the augmented training set $\mathcal{L} \cup \{(x_i, 0)\}_{i=n+1}^{n+n_0}$. This augmented set is used for training a standard random forest $\{h_i\}_{i=1}^{M}$.

We define:

$$\hat{o}(x) := \hat{p}(Y = 0|x) = \frac{1}{M} \sum_{i=1}^{M} \mathbf{1}\{h_i(x) = 0\}$$

For the implementation of the AL strategy presented in Section 6.2, only the rank order of $\hat{o}(x)$ among the unlabeled points is relevant. As shown in the previous chapter in Section 5.4.2.2, the practical influence of the parameter n_0 on the rank order is only minor. Therefore, n_0 is simply set to n in the experimental section.

6.3.3 Two-Class Learning

As soon as the training set $\mathcal{L}$ contains positive and negative points, we want a method that, for each query point x, returns an estimate $p(y|x)$ for the posterior class probabilities together with a statement regarding the consistency of x with $\mathcal{L}$ (C2). To this end, we can use the same idea as in the previous section, namely adding uniformly distributed reference data to the training set. The difference is that the artificial data leads to a three-class problem here.

As before, let $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{n}$ be the current training set and let the rectangle $\mathcal{R}$ cover $\mathcal{L}$ and all unlabeled points, i.e. $\{x_i\}_{i=1}^{n} \cup \{x \in B : B \in \mathcal{U}\} \subset \mathcal{R}$. We draw n_0 samples $\{x_i\}_{i=n+1}^{n+n_0}$ from the uniform distribution on $\mathcal{R}$ and label them as 0. The samples are combined to obtain a new training set $\mathcal{L} \cup \{(x_i, 0)\}_{i=n+1}^{n+n_0}$. This augmented set is used for training a standard random forest $\{h_i\}_{i=1}^{M}$.

In the following, two different kinds of posterior class probabilities need to be distinguished. Whereas the probabilities of the original two-class problem are referred to with the letter “p” (as defined in Section 6.2.1), the letter “q” is used for the three-class problem. It can easily be shown by employing Bayes’ theorem that

$$p(Y = y|x) = \frac{q(Y = y|x)}{q(Y = -1|x) + q(Y = 1|x)}, \quad y = -1, 1 \tag{6.3}$$

i.e., the posterior class probabilities of the two-class problem can easily be calculated from those of the three-class problem (see also Eq. (3.6)). It follows that the posterior class probabilities of the two-class problem can be estimated by

$$\hat{p}(Y = y|x) = \frac{\sum_{i=1}^{M} \mathbf{1}\{h_i(x) = y\}}{\sum_{i=1}^{M} \left[\mathbf{1}\{h_i(x) = -1\} + \mathbf{1}\{h_i(x) = 1\}\right]}, \quad y = -1, 1$$

We define $\hat{p}(Y = y|x) := 1/2$ if $\sum_{i=1}^{M} \left[\mathbf{1}\{h_i(x) = -1\} + \mathbf{1}\{h_i(x) = 1\}\right] = 0$.

In contrast to $\hat{p}(Y = y|x)$, there is no obvious choice for the definition of $\hat{u}(x)$. In accordance with the definition of $\hat{o}(x)$ above, we set

$$\hat{u}(x) = \hat{q}(Y = 0|x) = \frac{1}{M} \sum_{i=1}^{M} \mathbf{1}\{h_i(x) = 0\}$$

Finally, we comment on the choice of n_0. According to Eq. (6.3), $p(Y = 1|x)$ is independent of the class prior $p(Y = 0)$ and thus n_0. In practice, the "sphere of influence" of the training samples in $\mathcal{L}$ is slightly decreased by the additional reference data [110]; thus the variance of the estimate $\hat{p}(Y = 1|x)$ is increased and its bias lowered (see e.g. [81, chap. 2]). However, the influence of n_0 on $\hat{p}(Y = 1|x)$ is only minor. In contrast, $\hat{u}(x)$ highly depends on n_0. Thus, in addition to γ, the parameter n_0 governs the trade-off between exploration and exploitation (see Eq. (6.1)). To suppress one of the parameters, we propose to set $n_0 = n$ as in Section 6.3.2 and to govern the trade-off with the parameter γ. An advantage of this agreement is that the random forest can be trained with a relatively balanced training set, avoiding the problems associated with imbalanced data [35].

6.4 Technical Details

In order to apply the AL strategy presented in Section 6.2 to industrial optical inspection, two application-specific problems need to be addressed. The first one is:

1. How can images be represented in feature space?

As noted in Sections 6.1 and 6.2 and presented in detail below, we divide the images into m patches, where each patch is represented by a point in feature space. This raises the second problem:

2. How can image-wise classification be achieved if there are only patch-wise estimates for the posterior class probabilities?

These two questions are discussed in the following.

6.4.1 Image Representation

Generally speaking, we need a feature space representation of the images to train a classifier that can discriminate between defective and non-defective parts. Considering only the gray values of an image is typically insufficient and thus some neighborhood information are required. Further, as the appearance of the defects is not known a priori and as algorithmic solutions for defect detection are supposed to work without human interaction, the extracted features need to be sufficiently rich to detect defects of different sizes and orientations.

To that end, we use the image representation that is proposed in [180].[7] The images are decomposed into scale and orientation subbands using a steerable pyramid [169] whose basis functions are directional derivatives operators. 5 pyramid levels are constructed and fifth-order directional derivatives are used leading to 6 orientation bands. This results in 30 feature images.

However, the resolution of these images is not equal, e.g. for the data presented in Section 6.5.1, ranging from 512×512 in the first to 32×32 in the fifth level of the pyramid. Instead of adjusting the resolutions e.g. by interpolation techniques, the *original* image is divided into quadratic patches of 32×32 pixels and seven different statistical quantities of the filter responses are computed on each patch: minimum, maximum, median, mean, variance, kurtosis and entropy. This yields a feature vector of dimension $d = 210$ for each patch. Note that the number of values the statistical quantities for a patch are based on ranges from $32 \cdot 32 = 1024$ in the first to $2 \cdot 2 = 4$ in the fifth level of the pyramid. Note further that the statistical quantities at patch-level introduce valuable *non*-linear information for the characterization of the original image texture at the corresponding location without using non-linear filters. Another advantage is that representing patches instead of pixels in feature space comes along with a substantial data reduction. This is especially useful for AL where the feature values only need to be calculated once prior to the AL process, but where the model needs to be updated and posterior class probabilities and "outlyingness" need to be calculated in each iteration.

Since each of the feature vectors representing an image corresponds to a patch, the elliptic pixel-wise labels need to be adapted accordingly. Here, a point is positive if and only if the center of the corresponding patch is within the ellipse encircling the defect.

Finally, note that a grid of non-overlapping patches may suffer from boundary effects. As an extreme example, consider that all scratches in the training images run through patch centers, whereas the scratch in a test image runs along patch boundaries. To that end, we work with overlapping patches, where the overlap is half the patch size. For an image with a resolution of 512×512 pixels and a patch resolution of 32×32 pixels, we obtain $m = 31 \cdot 31 = 961$ patches per image.

6.4.2 Image Classification

According to the image representation introduced above, each point in feature space corresponds to a patch of an image. This means for a bag $B = \{x_i\}_{i=1}^m$ representing a test image that m different posterior estimates $\{\hat{p}(Y = 1|x_i)\}_{i=1}^m$ are obtained. These es-

[7] The features used for the experiments in Section 6.5 also have been precomputed and provided by the authors of [180].

timates need to be merged to an image-wise prediction, either to some defect prediction value[8] or at least to a crisp class assignment.

As posterior class probabilities of adjacent patches are correlated, more robust estimates can be obtained if salt and pepper noise structures are removed from the probability maps in a first step. Here, we use a 3×3 median filter to smooth the maps. The mask size is a compromise between utilizing spatial information and not smoothing away indications for small defects. Then, as we have no prior information about defect characteristics, in particular the defect size, we simply use the maximum value of the smoothed probability maps as defect prediction value for the whole image.

6.5 Empirical Evaluation

In this section, we first present the data used for the evaluation of the proposed AL strategy. Afterward, we present the corresponding experimental results.

6.5.1 Data

For the statistical evaluation of the presented AL approach, we use 6 synthetic image data sets that cover a wide range of defects in industrial quality control on different textured surfaces. The images have been created under the supervision of Matthias Wieler for a defect detection contest at the annual conference of the DAGM[9] in 2007 [180]. They have been published on the internet[10] to make them available to the participants of the contest and to establish a benchmark data set for defect detection algorithms. The benchmark data has, for example, been used in [146].

Figs. 6.4 and 6.5 show two example defect images of each data set together with their elliptic label. The latter has been provided by a human annotator according to the procedure introduced in Section 6.1.

Each data set contains 1150 gray value images, of which 150 show defects. The spatial resolution of the images is 512×512, the gray value resolution is 8 bit. We have randomly partitioned each data set into a training and a test set of equal size.

6.5.2 Results

In this section, we present empirical results for the proposed 2-step AL strategy. As mentioned in the introduction of this chapter, to the best of our knowledge, this is the

[8] This does not need to be a probability in the strict sense but can be a real value on a “defect scale”.

[9] Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., http://www.dagm.de

[10] http://klimt.iwr.uni-heidelberg.de/dagm2007/prizes.php3

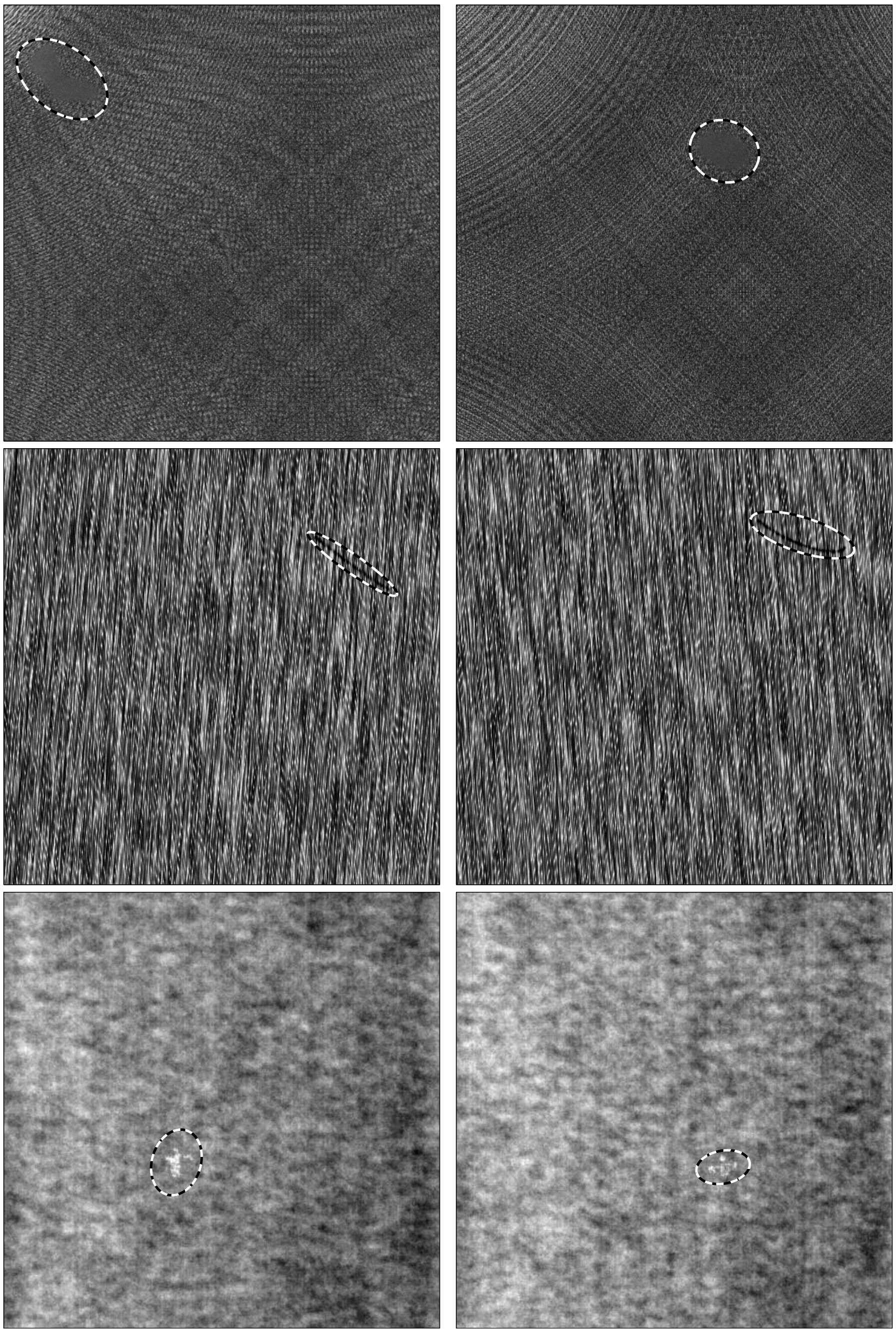


Figure 6.4: Two example defect images of data set 1 (top), 2 (middle) and 3 (bottom). Each data set is affected by a specific kind of defect. The ellipse in each image has been drawn by a human annotator.

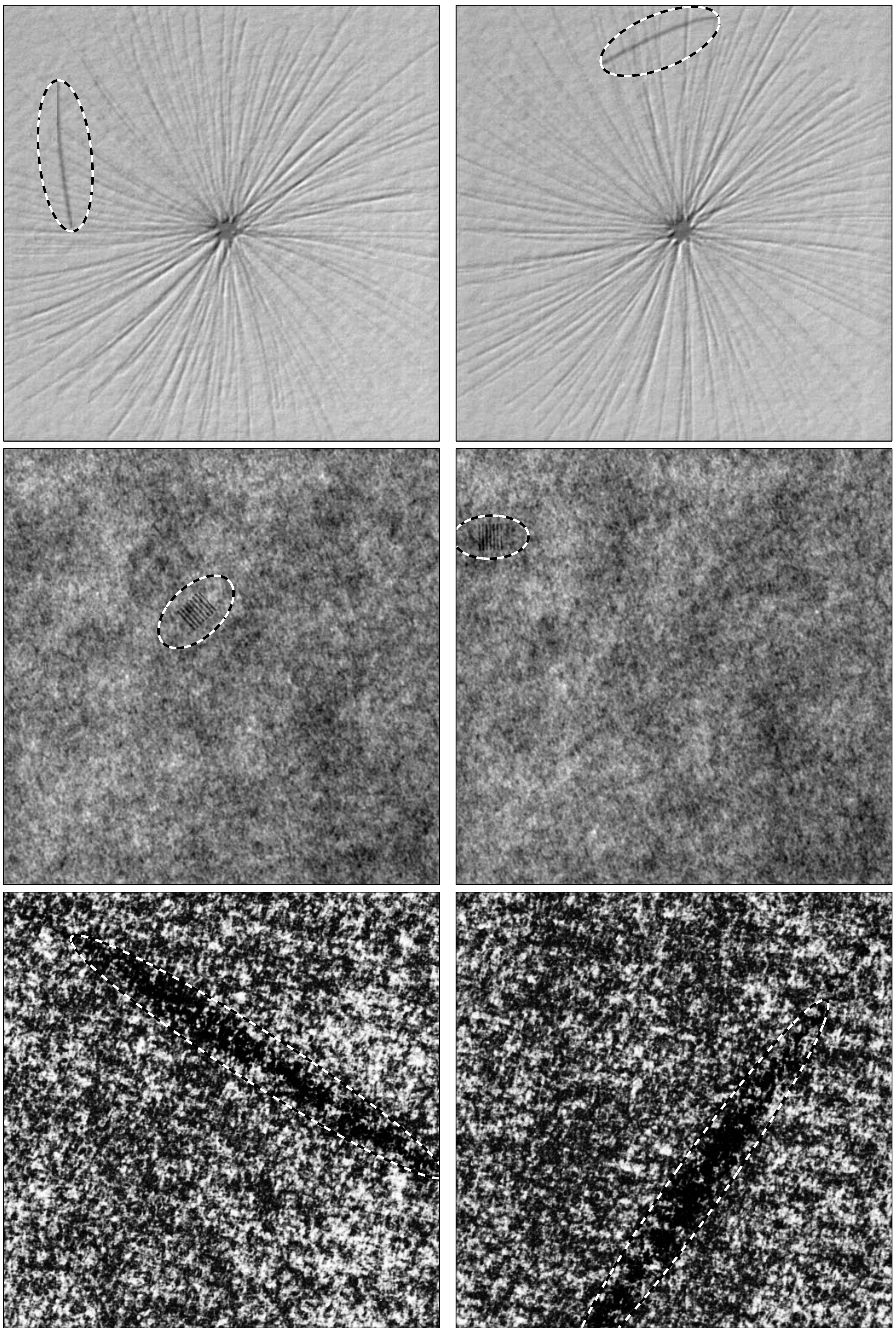


Figure 6.5: Two example defect images of data set 4 (top), 5 (middle) and 6 (bottom). Each data set is affected by a specific kind of defect. The ellipse in each image has been drawn by a human annotator.

first AL strategy for the setting considered here. Thus, we compare the performance of the proposed strategy to random sampling to evaluate if there is at all a benefit from AL. To get a better understanding of the 2 steps of the proposed strategy, we also compare to“1-step AL”, where only the query step is performed and the elimination step is omitted.

In all experiments, the AL process is initiated by drawing randomly $b_1 = 40$ from the *negative* bags in $\mathcal{U}$ that make up a first training set $\mathcal{L}$ of $b_1 m = 40 \cdot 961 = 38{,}440$ data points. We disallow positive bags in the first iteration to prevent lucky strikes that spare the way through the one-class learning part of the process. In each subsequent iteration of the process, $b_2 = b_3 = 5$ bags are selected from the remaining bags in $\mathcal{U}$. They are chosen randomly in case of random sampling, whereas in 2-step and 1-step AL, the bags are chosen according to the procedure described in Section 6.2.2.2. The parameter γ that governs the trade-off between exploration and exploitation is set to 1.[11] In 2-step AL, the elimination step is then performed as described in Section 6.2.2.3. The cardinal number of $B_{\mathcal{L}}$ equals $\lfloor 0.05 \cdot 961 \rfloor = 48$ for each selected bag B. In 1-step AL and random sampling, all positive points in B are added to $\mathcal{L}$ and a random sample of “48 minus the number of positive points in B” from the remaining negative points.

To gain a first insight into the proposed 2-step AL approach, we look at the development of the point-wise classifier output for a bag representing a test image of data set 5. This is shown in Fig. 6.6. After the first labeling iteration (40 bags), there are only negative points in $\mathcal{L}$ and we recur to one-class learning. The output of the classifier is the “outlyingness” measure $\hat{o}(x)$ as shown in the upper left panel of the figure. It can be observed that $\hat{o}(x)$ is relatively large for many positive points. This shows in particular that $\hat{o}(x)$ is a useful measure for identifying positive bags for subsequent labeling.

Indeed, in this example, positive bags have been found in $\mathcal{U}$. Accordingly, the output of the classifier after the second iteration (45 bags) of the process is a posterior prediction $\hat{p}(Y = 1|x)$ and its uncertainty $\hat{u}(x)$. As the number of positive points in $\mathcal{L}$ is small at the beginning, all posterior predictions for positive points are smaller than 0.5, a known problem when learning with imbalanced data [35]. In the subsequent iterations (50-100 bags), the number of training points in the neighborhood of the positive points increases, indicated by smaller values for the uncertainty $\hat{u}(x)$. At the same time, the posterior prediction for the positive points increases. Note that the posterior prediction of a few positive points is close to 0, even after 100 labeled images. These are points with an incorrect label (non-defective patches within the ellipse).

Next, we compare the classification performance of the proposed 2-step AL approach to 1-step AL and random sampling. After each iteration of the AL process, a random forest classifier is trained on the current training set $\mathcal{L}$ and evaluated on the test data

[11] As the defects in a specific DAGM data set look quite alike, the results are not sensitive to varying γ. The parameter may be set to a larger value if defects are very diverse.

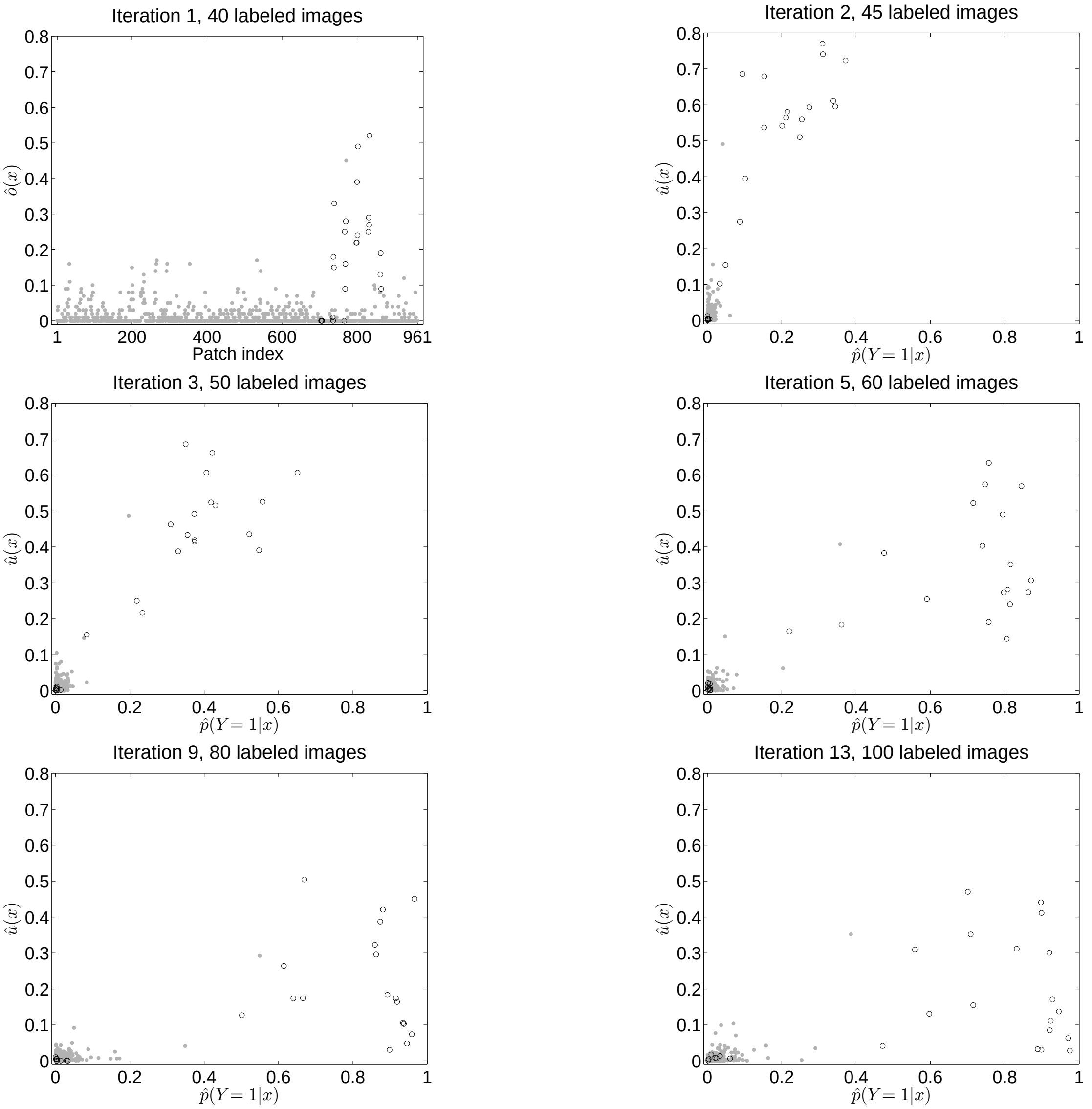


Figure 6.6: Development of the classifier output for the points of a positive bag representing a test image of data set 5. Every point is depicted as a circle; negative points are filled and gray; positive ones are open and black. As long as there are only negative points in the training set $\mathcal{L}$ (upper left panel), the classifier returns a measure $\hat{o}(x)$ for "outlyingness" only. As soon as $\mathcal{L}$ contains both positive and negative points, the classifier returns a posterior prediction $\hat{p}(Y = 1|x)$ and a measure $\hat{u}(x)$ for the uncertainty in this prediction. It can be observed that the capability of the classifier to distinguish between positive and negative points increases from iteration to iteration. The positive points within the cloud of negative ones are incorrectly labeled points.

according to the procedure presented in Section 6.4.2. As the accuracy is an inappropriate performance measure if the data is imbalanced[12] [147], we use the area under the receiver operating characteristic (ROC) curve (AUC) [59]. An AUC of 1 corresponds to perfect separability of the two classes with respect to posterior predictions.

The results for all 6 data sets, averaged over 6 runs with different random seed, are shown in Fig. 6.7. Each plot shows the development of the AUC depending on the number of labeled images/bags. For 5 data sets (1–3, 5 and 6), 2-step and 1-step AL achieve much steeper learning curves at the beginning and, additionally, attain an AUC of 1 or close to one after fewer iterations than random sampling. The only exception is data set 4, where random sampling is more efficient at the beginning, achieving an acceptable classification performance with about 25 bag labels less. The reason is that the defects of the images of data set 4 are relatively similar to the background texture and are thus relatively difficult to identify for the outlier detection/one-class learning algorithm. As soon as examples of defects are available, the performance of the two AL approaches rapidly improves and becomes better than that of random sampling.

The two AL approaches perform very similar in the experiments above and the elimination step of the 2-step strategy seems dispensable at first sight. The reason is that random forests are known to be very robust to label noise [24] and the difference does not become visible as long as the number of wrong labels is sufficiently small. To investigate this in more detail, we enlarge the elliptic labels. To be more precise, the labels are dilated by a disc-shaped structuring element of radius 128 pixels. The corresponding results are shown in Fig. 6.8. Whereas the results of the 2-step AL approach remain comparable to these using the original weak labels, the results of 1-step AL deteriorate for data sets 2 and 6; the AUC is not even close to 1 for data set 6 after labeling 200 images.

6.6 Conclusions

In this chapter, we have presented a novel AL approach that responds to three challenges in industrial optical inspection: weak labels, bundled label query and imbalanced classes. Each iteration of the proposed strategy consists of two steps: in the query step, the images for subsequent labeling by a human annotator are selected and in the elimination step, a subset of the patch-wise weak labels obtained for each selected image is chosen. The strategy has been instantiated by extending standard random forests.

Based on 6 publicly available benchmark data sets, we have shown that AL techniques

[12] Consider a data set where positive and negative samples appear at a ratio of 999:1. Then the accuracy of the primitive classifier that always assigns the negative class is 99.9%. Despite the “good” performance, this trivial solution obviously is unsatisfactory.

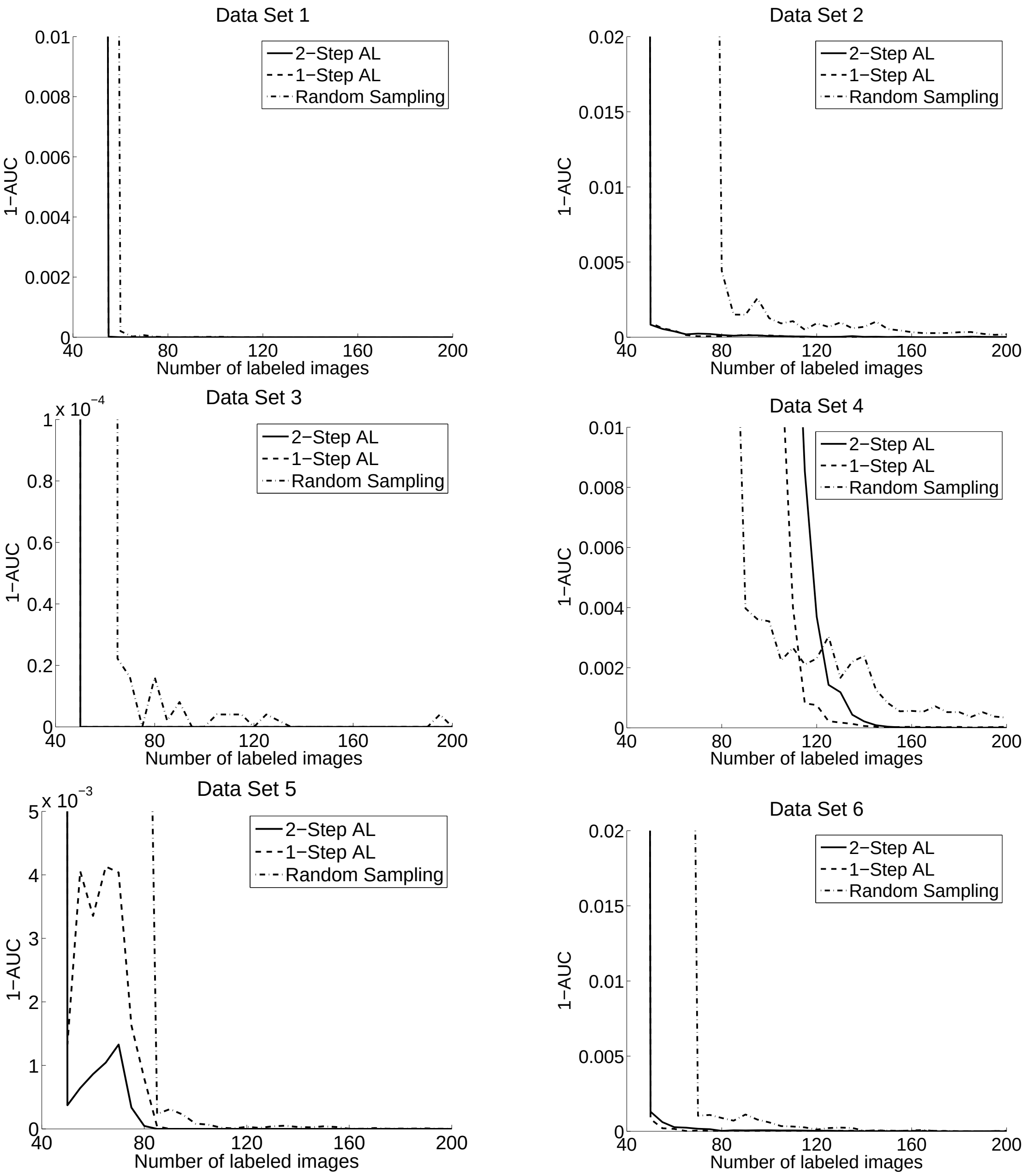


Figure 6.7: Performance comparison of the proposed 2-step AL approach with 1-step AL and random sampling. Each plot shows the development of the area under curve (AUC) depending on the number of labeled images for one out of 6 different data sets. For most of the data sets, the two AL approaches achieve steeper learning curves and earlier attain perfect separation of the two classes. Note the different scales of the y-axes.

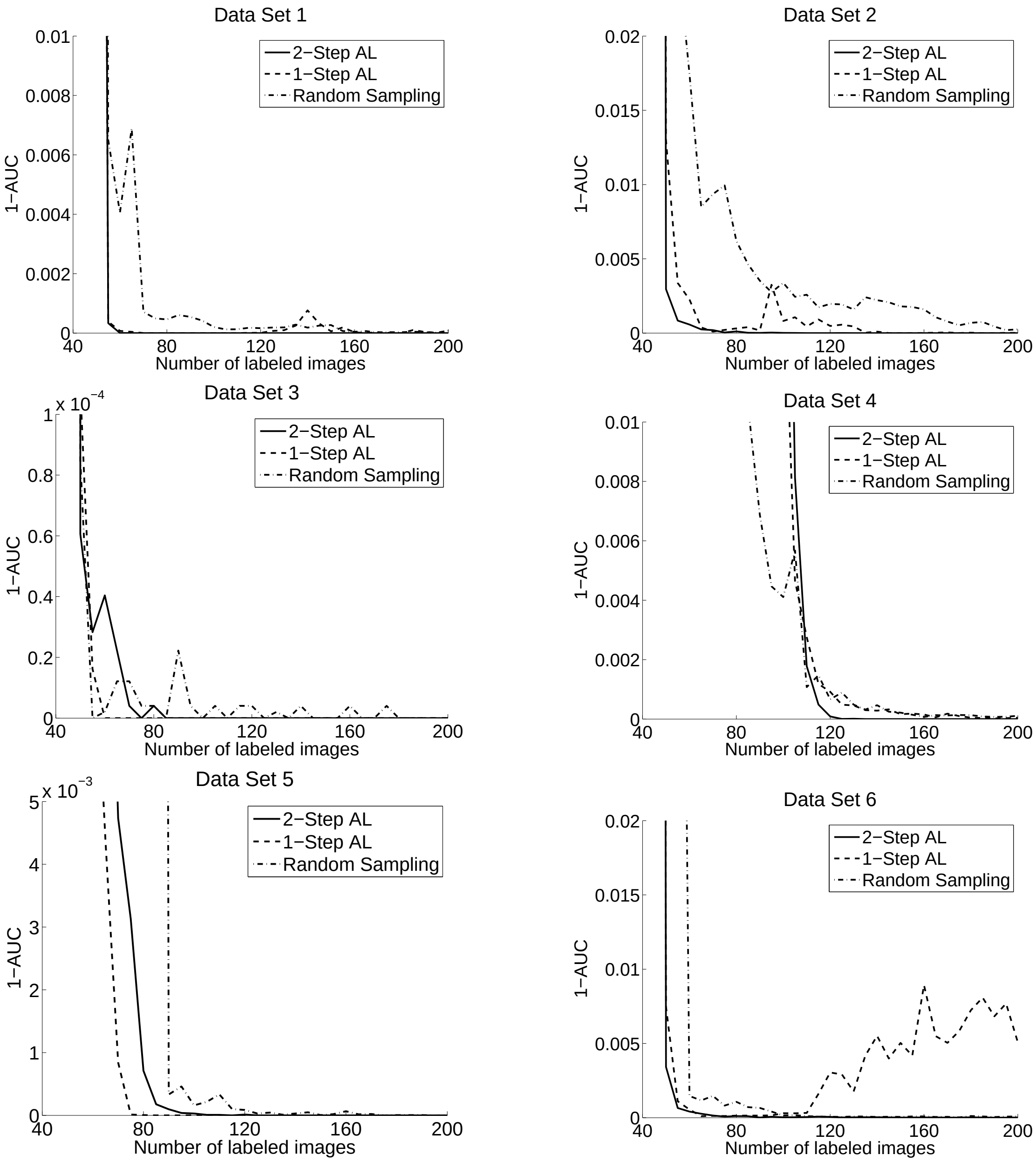


Figure 6.8: Performance comparison of the proposed 2-step AL approach with 1-step AL and random sampling after having dilated the labels by a disc-shaped structuring element of radius 128. Each plot shows the development of the area under curve (AUC) depending on the number of labeled images for one out of 6 different data sets. Compared to the results with the original labels in Fig. 6.7, the performance of 1-step AL deteriorates for data sets 2 and 6. Note the different scales of the y-axes.

indeed can reduce the labeling effort required for achieving a certain classification performance (compared to random sampling). The key to this improvement is the query step. It strives to preserve a balance between exploring the feature space and seeking out positive labels in order to compensate for the underrepresented class of defects. The additional elimination step is appropriate if the employed classifier is not sufficiently robust to label noise incurred by weak labeling.

One drawback of the 6 benchmark data sets—particularly when using them for the evaluation of AL methods—is that each of them contains one specific kind of defect only. Here, the parameter γ thus has not been optimized but has simply been set to 1.

7 Gaussian Process Classification

The aim of this chapter is to compare three different methods for binary classification with an underlying Gaussian process with respect to theoretical consistency and practical performance. Two of the inference schemes, namely classical indicator kriging and simplicial indicator kriging, standard methods in geostatistics, are analytically tractable and fast. However, these methods rely on simplifying assumptions which are inappropriate for categorical class labels. A consistent and previously described model extension involves a doubly stochastic process. There, the unknown posterior class probability $\pi(\cdot)$ is considered a realization of a spatially correlated Gaussian process with function values squashed to the unit interval, and a label at position x is considered an independent Bernoulli realization with success parameter $\pi(x)$. Unfortunately, inference for this model is not known to be analytically tractable. In this chapter, we propose a new computational scheme for the inference in this doubly stochastic model, namely the "Doubly Stochastic Gaussian Quadrature". The method is analytical up to a final step where integration must be carried out numerically. For the comparison of practical performance, the methods are applied to storm forecasts for the Spanish coast based on wave heights in the Mediterranean Sea. While the error rate of the doubly stochastic models is slightly lower, their computational cost is much higher.

7.1 Introduction

Gaussian process *regression* has been introduced to the field of machine learning by Williams and Rasmussen [184]. Before, it had long been known in the fields of geostatistics (under the name of "kriging" [37]) and in signal processing [182]. Its popularity is due to its flexibility, mathematical tractability, its natural Bayesian interpretation and success in a wide range of applications [70, 148, 108]. In Gaussian process regression, the observed outputs of the points in feature space are assumed to arise from the realization of a Gaussian process with or without Gaussian noise. As briefly explained in Section 7.2.1, an approximation or interpolation for all points in feature space (given the observed data and assumptions on the mean and covariance structure of the Gaussian process) is then obtained from the best linear unbiased estimator.

Unfortunately, inference is more complicated in *classification*. Whereas a Gaussian prior can be combined with a Gaussian likelihood in the case of regression (resulting in a simple computational scheme revolving around a linear system of equations), a

Gaussian likelihood is obviously inappropriate for discrete class labels [149].

In this chapter, we compare three different approximation schemes for *binary* Gaussian process classification: classical indicator kriging (CIK) [90], simplicial indicator kriging (SIK) [175] and the doubly stochastic Gaussian quadrature (DSGQ). The latter is a new computational scheme for a model that has previously been studied in environmental applications [51] and in machine learning [183]. This model is motivated and presented in the following paragraphs.

Let $\mathcal{T} = \{(x_i, y_i), i = 1, \ldots, n\}$ be a training set sampled from a random vector (X, Y), where $y_i \in \{0, 1\}$ denotes the binary class label of a feature vector $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$. Let $\mathbf{y} \in \{0, 1\}^n$ and $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the vector of labels and matrix of feature vectors, respectively. The goal is to predict the posterior class probability $p(Y = y_* | x_*) \equiv p(Y = y_* | \mathbf{X}, \mathbf{y}, x_*)$ of the unknown label y_* at a point x_* given the training set.[1]

The classical approach [90] is to compute the posterior probabilities by fitting the binary labels directly, without regard to the fact that the y_i cannot be normally distributed. This approximation is called (classical) indicator kriging and has a long record of successful applications. However, there are two main problems with CIK: First, it quite often delivers probabilities that are smaller than 0 or larger than 1, and second, the order relation of probabilities is violated. The latter means, as explained in more detail in Section 7.2.2, that the difference on the real line is not adequate to express distances between probabilities.

These drawbacks are tackled by SIK [175]. There,

1. the probability $p(Y = 1|x)$ is considered the function value $\pi(x)$ of an unobservable realization $\pi(\cdot)$ of a Gaussian process "squashed" to the open unit interval $(0, 1)$ as defined in the next section.

But, as will be discussed in Section 7.2.4, SIK still makes simplifying assumptions that may not be satisfactory in general. In particular, $\pi_i := \pi(x_i)$ can only be either p or $1 - p$, $p \in (0, 0.5)$, depending on which of the two classes is observed at the training location x_i, regardless of any other observations in the vicinity. This is contrary to intuition. For example, consider one point in feature space and its immediate neighbors, and two possible scenarios: first, that a "success" has been observed at all these points; and secondly, that "success" has been observed only at the central point and "failure" at all others. According to SIK, the posterior probability at the central point would be the same in both scenarios. The model must hence be extended such that we fully distinguish between an observed label and its estimated probability:

2. the observed labels y_i are *conditionally* independent realizations of Bernoulli distributions with parameters $\pi_i = p(Y = 1|x_i)$, i.e. $Y|\pi_i \sim \mathcal{B}ern(\pi_i)$.

[1] Note that we slightly deviate here from the notation of the rest of this thesis. This is to simplify the derivations later on in Section 7.4.

A graphical representation of this doubly (1. and 2.) stochastic model is shown in Fig. 7.1.

In summary, CIK and SIK are based on model approximations that may be inconsistent with some or all characteristics of a classification setting, but that are linear in output measurements and thus analytically tractable and fast. In contrast, the doubly stochastic model is consistent, but predictive inference needs to be approximated.

Many approximation schemes have been proposed, among them Laplace's method [183], the integrated nested Laplace approximation [154], Markov chain Monte Carlo (MCMC) approximations [133, 51], expectation propagation [126], the cavity TAP approximation [137] and a variational approximation [71]. An excellent review is presented in [100]. In the field of geostatistics, modifications of CIK are also abundant [91, 30, 171, 141].

The contribution of this chapter is twofold. First, we compare CIK, SIK and the doubly stochstic model with respect to theoretical consistency and practical performance. Second, for this comparison, a new approximation scheme for the doubly stochastic model is presented, the doubly stochastic Gaussian quadrature (DSGQ). The method is analytical up to a final step where optimization or integration must be performed numerically. This extends the insight into the doubly stochastic model and may form the basis for future research.

The next section reviews the most typically used method in the geostatistical community, CIK, as well as an alternative based on the Aitchison geometry of the unit interval, SIK. The underlying geometry is also presented in Section 7.2. Section 7.3 introduces a distribution on the unit interval, compatible with this geometry, that plays the role of a prior distribution of the vector of probabilities of interest. Section 7.4 then uses this distribution to derive an estimator for the unknown probabilities that is based on a doubly stochastic Gaussian process.

An experimental comparison of all methods is presented in Section 7.5. To that end, a given forecast of wave heights in the Mediterranean Sea is classified in two conditions: *Eastwind-storm* (called *Llevant* in Catalan) and *any other situation* (either calm or any other of the dominant windstorms of this region) are the two possible labels. In this setting, the feature vector $x \in \mathbb{R}^d$ consists of the values at d predefined pixels of a forecast map.

7.2 Classical and Simplicial Indicator Kriging

In this section, we briefly review two approximation methods for Gaussian process classification that do not consider a Bernoulli distribution for the observed labels, namely classical and simplicial indicator kriging. We first describe simple kriging which is then applied to predict posterior class probabilities in a classification setting.

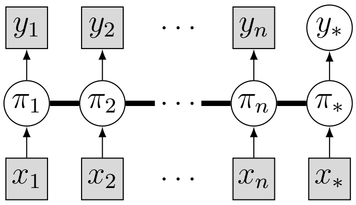


Figure 7.1: Graphical representation of the doubly stochastic model. Observed variables are shaded squares, circles represent unknowns. The thick lines indicate a fully connected graph. The first stochastic layer is given by the posterior class probabilities $\pi_i, i = 1, \dots, n$ and π_* that are considered function values of a squashed Gaussian process, the second layer is given by the observed labels that are Bernoulli distributed with parameter π_i.

7.2.1 Simple Kriging

Let $\{(x_1, \pi_1), \dots, (x_n, \pi_n)\}$ be n pairs of sampling points x_i and outputs $\pi_i = \pi(x_i), i = 1, \dots, n$. In case of simple kriging it is assumed that $\pi(x)$ is a function value of a realization of a Gaussian process with known mean and covariance structure $\mathrm{C}(x_i, x_j) = \mathrm{Cov}[\pi(x_i), \pi(x_j)]$, i.e. the joint distribution of any subset of observed or unobserved points is a multivariate normal. Assuming a zero mean, the estimate for the function value at an arbitrary point x_* in feature space is of the form (see e.g. [37, 149])

$$\hat{\pi}_* = \hat{\pi}(x_*) = \sum_{i=1}^{n} \lambda_i(x_*)\pi(x_i) \tag{7.1}$$

i.e. the simple kriging estimator is a linear combination of the function values at the sampling points. The coefficients $\lambda_i, i = 1, \dots, n$, depend on the position of prediction and are obtained maximizing the well-known normal conditional density (see e.g. [149])

$$p(\pi_*|\boldsymbol{\pi}, \mathbf{X}, x_*) = \frac{1}{\sqrt{\tau\left(\sigma_*^2 - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}\right)}} \exp\left(-\frac{1}{2}\frac{\left(\pi_* - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\pi}\right)^2}{\sigma_*^2 - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}}\right)$$

where $\boldsymbol{\pi} = (\pi_1, \dots, \pi_n)^T$, $\boldsymbol{\Sigma}_{ij} = \mathrm{C}(x_i, x_j)$, $[\boldsymbol{\sigma}]_i = \sigma_i = \mathrm{C}(x_i, x_*)$, $\sigma_* = \mathrm{C}(x_*, x_*)$ and $\tau := 2\pi$, with π being the mathematical constant in this case.[2] This gives kriging weight

$$\lambda_i(x_*) = [\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}]_i$$

[2] We use the definition $\tau := 2\pi$ here and in the following to avoid confusion between the mathematical constant π and the symbols $\boldsymbol{\pi}$, π_i, $\pi(\cdot), \dots$ The latter are used to be consistent with the notation introduced in Chapter 3.

The matrix $\mathbf{\Sigma}$ is invertible if the covariance function is strictly positive definite and if all the sampling points are distinct. Note that the coefficients $\lambda(x_*)$ do not sum up to one in general, and can even be negative. For further details on simple kriging, other kriging "flavors" or examples of covariance functions, see e.g. [37] and [149].

7.2.2 Classical Indicator Kriging

The easiest possibility to predict posterior class probabilities at a point x_* in feature space is classical indicator kriging (CIK) [90]. There, the binary class labels $y_i \in \{0,1\}$ are treated as function values, i.e. $\pi_i := y_i, i = 1, \ldots, n$, and the probability of success is directly given by the simple kriging estimate $\hat{\pi}_*$.

However, CIK has two major drawbacks. First, although the data $y_i \in \{0,1\}$, it is not guaranteed that the interpolation $\hat{\pi}_* \in (0,1)$, which is necessary in order to interpret it as a probability. Second, the order relation of probabilities is violated, i.e. distances between probabilities are not represented accurately by their difference on the real line. Consider the following example of two pairs of probabilities: $(0.001, 0.01)$ and $(0.501, 0.51)$. In the first case, the second probability is ten times higher, whereas in the second case, the probabilities are almost equal; but the actual distances on the real line are 0.009 in both cases. This suggests that a change of geometry may be adequate, which is presented in the next section.

7.2.3 Geometry in the One-dimensional Simplex $\mathbb{S}^2$

Consider the line segment $\{(a, 1-a), a \in (0,1)\} \subset \mathbb{R}^2$, which is equal to the one-dimensional positive simplex $\mathbb{S}^2$. The simplex $\mathbb{S}^2$ is useful to represent the probability of a certain event together with its complementary probability because the components of an element of $\mathbb{S}^2$ always add up to 1. Moreover, it has a Euclidean vector space structure, called *Aitchison Geometry*, if it is endowed with the following three operations [143, 19]. There, $\mathcal{C}(a) := (a_1/(a_1+a_2), a_2/(a_1+a_2))$ divides each component of a vector by the sum of its components to ensure the closure under addition and scalar multiplication.

(i) Vector addition: $a \oplus b := \mathcal{C}(a_1 b_1, a_2 b_2)$, representing addition of information following Bayes' theorem

(ii) Scalar multiplication: $\lambda \odot a := \mathcal{C}(a_1^\lambda, a_2^\lambda), \lambda \in \mathbb{R}$

(iii) Scalar product: $\langle a, b \rangle := 1/c_0^2 \ln(a_1/a_2) \ln(b_1/b_2)$

The constant c_0^2 is a scaling parameter. As explained in detail in Section 7.3, it is intimately related to the variance of the normal distribution on the hypercube. It follows immediately from the above definitions that the additive neutral element of $\mathbb{S}^2$ is

$\mathcal{C}(1,1) = (1/2, 1/2)$ and the inverse element of $a = (a_1, a_2)$ is (a_2, a_1). Moreover, we automatically obtain an algebraic definition of the distance in $\mathbb{S}^2$:

$$d(a,b) = \|a \ominus b\| = \sqrt{\langle a \ominus b, a \ominus b\rangle} = \frac{1}{c_0}\sqrt{\left(\ln\left(\frac{a_1}{a_2}\right) - \ln\left(\frac{b_1}{b_2}\right)\right)^2} \tag{7.2}$$

where subtraction is defined by addition with the inverse element. The norm of a vector (a_1, a_2) in this geometry is $\|a\| = \sqrt{\langle a, a\rangle}$.

As in every Euclidean vector space we can choose an orthonormal basis—which, in this case, consists of one vector only: $e_b = \mathcal{C}(\exp(c_0), 1)$. In the *coordinate representation*, each element $a \in \mathbb{S}^2$ is uniquely represented with respect to the chosen basis:

$$\alpha = \langle a, e_b\rangle = \frac{1}{c_0}\ln\left(\frac{a_1}{1-a_1}\right) \tag{7.3}$$

Conversely, the element a can be computed from its coordinate representation by scalar multiplication: $a = \alpha \odot e_b = \mathcal{C}(\exp(c_0\alpha), 1) = (a_1, 1-a_1)$. The mapping from $\mathbb{S}^2$ to $\mathbb{R}$ assigning a coordinate to each point is an isomorphism. Furthermore, all points in $\mathbb{S}^2$ are uniquely determined by their first component, so we can identify a point a_1 on the real interval $(0,1)$ with the point $(a_1, a_2) = (a_1, 1-a_1)$ on $\mathbb{S}^2$ and hence the interval $(0,1)$ with the simplex $\mathbb{S}^2$. This leads to an isomorphism from the interval $(0,1)$ to $\mathbb{R}$.

7.2.4 Simplicial Indicator Kriging

The main drawbacks of CIK, stated in Section 7.2.2, are tackled by SIK [175]. This method is based on the realization that there is no need to establish an identity between a probability $\pi \in (0,1)$ and its representation ϕ on the real line. They are better connected by the logit transformation:

$$\phi = \ln\frac{\pi}{1-\pi} \tag{7.4}$$

Note the correspondence between Eqs. (7.3) and (7.4). The constant $1/c_0$ is omitted here because it cancels out in the final estimate π_* of SIK.

The simplicial kriging estimate is obtained in four steps:

1. estimate $\pi_i = p(Y=1|x_i)$, the probabilities of success at each sample (of the training set); many estimation methods are possible [175], e.g. a Bayesian estimate combining a Jeffreys' prior with the observed class likelihood, which would yield $\hat{\pi}_i = 3/4$ if a success is observed at x_i, and $\hat{\pi}_i = 1/4$ otherwise;

2. get the logistic transformation of these estimates; being a log-ratio, this implies that extreme values of $\hat{\pi}_i = 1$ or $\hat{\pi}_i = 0$ should be avoided in the preceding step;

3. apply simple kriging to the logistic-transformed estimates $\hat{\phi}_i = \ln(\hat{\pi}_i / 1 - \hat{\pi}_i)$ to obtain an interpolation ϕ_* at an unclassified sample x_*,

4. undo the logistic transform, to obtain an interpolated probability $\hat{\pi}_* = \exp(\hat{\phi}_*)/(1 + \exp(\hat{\phi}_*))$.

The rationale behind SIK is to build the linear combination in Eq. (7.1) using the operations on the simplex explained in the preceding Section 7.2.3. Thus, SIK is an interpolation or approximation technique for probabilities within the framework of *squashed* Gaussian processes, as will be described next.

It is also shown in [175] that, if the estimates $\hat{\pi}_i$ are just $1 - p$ or p, $p \in (0, 0.5)$, wherever a success, respectively a failure is observed, results get actually very simple. In this simplified situation, the estimate from the latter can be derived from that of the former, denoted here as π_*^{CIK}, by

$$\pi_* = \operatorname{logit}^{-1} \left(2 \ln \frac{1-p}{p} \cdot \left(\pi_*^{CIK} - 0.5 \right) \right) . \tag{7.5}$$

Though Eq. (7.5) is an interesting way of “recycling” old, inconsistent CIK results into valid probabilities, estimating π_i by two values only (namely p and $1-p$) still is a gross simplification.

But the main problem of SIK is its inability to “transfer information” between labeled points in the first step, as detailed in the thought experiment specified in the introduction of this chapter. SIK is unable to deliver this result, because $\pi(x_i)$ is estimated separately at each point x_i, even in the presence of a nugget effect[3].

7.3 The Normal Distribution on the Unit Hypercube

In this section, we take the Euclidean vector space structure on the interval $(0, 1)$ given in Section 7.2.3, and define the normal distribution on the unit hypercube $(0, 1)^n$. This distribution serves as prior distribution for two of the methods for Gaussian process classification considered in this chapter, namely SIK (presented in the previous section) and DSGQ (introduced in Section 7.4).

The transformation induced by the isomorphism presented in Section 7.2.3 maps the conventional normal distribution defined on the real line to the interval $(0, 1)$:

Definition 7.1. *A random variable Z is said to be normally distributed on $(0, 1)$, denoted $Z \sim \mathcal{N}_{(0,1)}(\mu, \sigma^2)$, if its coordinate representation* (7.3) *is normally distributed on $\mathbb{R}$ with mean μ and variance σ^2. [142]*

[3] For an explanation, see e.g. [37].

It follows that the random variable Z has Lebesgue density

$$g(z|\mu,\sigma^2) = \frac{1}{c_0 z(1-z)} \frac{1}{\sqrt{\tau\sigma^2}} \exp\left(-\left(\frac{1}{c_0}\ln\left(\frac{z}{1-z}\right)-\mu\right)^2 / \left(2\sigma^2\right)\right) \tag{7.6}$$

$$= \frac{1}{z(1-z)} \frac{1}{\sqrt{\tau c_0^2\sigma^2}} \exp\left(-\left(\ln\left(\frac{z}{1-z}\right)-c_0\mu\right)^2 / \left(2c_0^2\sigma^2\right)\right), \tag{7.7}$$

$$z \in (0,1), \mu \in \mathbb{R}, \sigma \in \mathbb{R}^+$$

where the first factor in (7.6) comes from measure theory and compensates for the unfamiliar definition (7.2) of the distance in $\mathbb{S}^2$. Fig. 7.2 shows the probability density functions for $c_0 = 1$ and varying values of μ and σ^2.

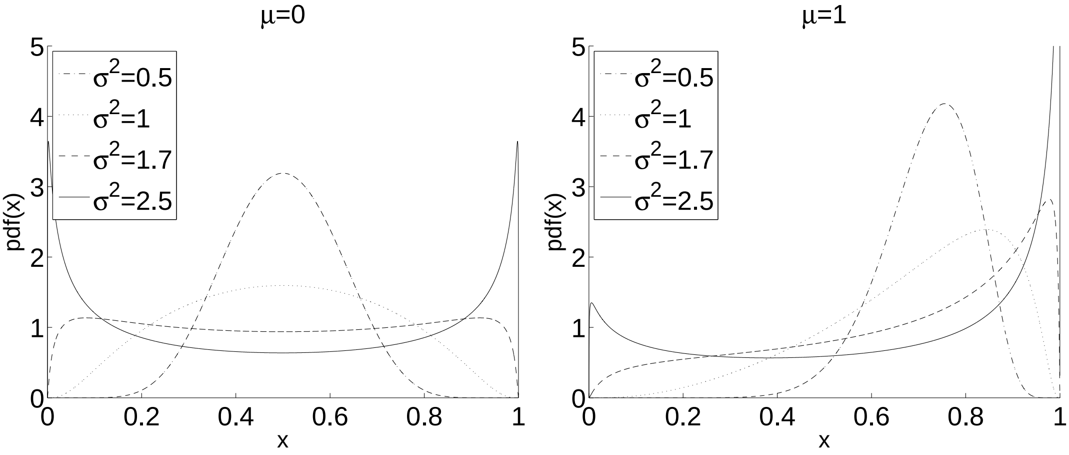


Figure 7.2: The normal distribution in $\mathbb{S}^2$ for different parameter values. For $\mu = 0$ (left panel), we obtain a symmetric density function around 0.5 ($0.5 \in (0,1)$ has coordinate representation $0 \in \mathbb{R}$). The bigger the variance σ^2 the more probability mass is concentrated near the boundaries of the interval. In contrast to the usual normal distribution, the density function is apparently not symmetric for $\mu \neq 0$ (right panel). The expectation value of this distribution converges to 1 for $\mu \to +\infty$, and to 0 for $\mu \to -\infty$.

The distribution of the latent variable at a single position in feature space lives on the interval $(0,1)$. The joint distribution of several variables—which will typically be dependent—then lives on the Cartesian product of these line segments, i.e. on the hypercube $(0,1)^n$.

Definition 7.2. *A random vector* **Z** *is normally distributed on* $(0,1)^n$*, denoted* **Z** $\sim \mathcal{N}^n_{(0,1)}(\boldsymbol{\mu},\boldsymbol{\Sigma})$*, if its coordinate representation* (7.3) *is multivariate normally distributed on* $\mathbb{R}^n$ *with mean* $\boldsymbol{\mu}$ *and covariance matrix* $\boldsymbol{\Sigma}$*.*

If we squash a Gaussian process to the unit interval according to the inverse of Eq. (7.3), its finite-dimensional distributions are normally distributed in the unit hypercube.

Remark 7.3. *Note that σ^2 and c_0 are intimately related and c_0 actually becomes a scaling parameter, or as was already mentioned, the units of the problem. This can be easily inferred from its behavior in term* (7.7) *for the one-dimensional distribution and is particularly evident for $\mu = 0$. In this case, σ^2 and c_0 become equivalent parameters. These considerations carry over to the multivariate case in Definition 7.2, where the multiplication of c_0 by a constant can be compensated by adapting $\mathbf{\Sigma}$ accordingly.*

Finally, note that the multivariate normal in the hypercube is not the only possible choice to model the prior distribution of a probability random field. Another approach not pursued here is using copulas instead [96].

7.4 Doubly Stochastic Gaussian Process

We introduce the doubly stochastic model in Section 7.4.1. The method presented in the subsequent section, namely DSGQ, is based on these model assumptions and is an estimator for the unknown posterior class probability $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*)$.

7.4.1 Posing the Model

Let us from now on use the coordinate representation $\phi_i \in \mathbb{R}$ for $\pi_i \in (0, 1)$ as introduced in Section 7.2.3,

$$\phi_i = \frac{1}{c_0} \log\left(\frac{\pi_i}{1 - \pi_i}\right) \Leftrightarrow \pi_i = \frac{e^{c_0 \phi_i}}{1 + e^{c_0 \phi_i}} \tag{7.8}$$

In the real coordinate space we can perform the usual Bayesian inference for regression without any restrictions, warranted by the *principle of working on coordinates* [142].

Recall that our goal is to predict the probability distribution of the unknown label Y at a point x_*, given the training set $\mathcal{T}$. In Section 7.1, we have introduced the two following model assumptions:

1. the probability $p(Y = 1|x)$ is considered the value $\pi(x)$ of an unobservable realization $\pi(\cdot)$ of a Gaussian process squashed to the unit interval,

2. the observed labels y_i are *independent* realizations of Bernoulli distributions with parameters $\pi_i = p(Y = 1|x_i)$, i.e. $y|\pi_i \sim \mathcal{B}ern(\pi_i)$.

This two-layer model can be successfully tackled in a Bayesian framework.

The second assumption implies that the likelihood of a sampled label vector **y** is

$$p(\mathbf{y}|\boldsymbol{\pi}) = \prod_{i=1}^{n} p(y_i|\boldsymbol{\pi}) = \prod_{i=1}^{n} p(y_i|\pi_i) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{1-y_i}$$

According to our first prior assumption, we may consider the unobserved success probability $p(Y = 1|x)$ to follow normal distribution on the hypercube, as given by Definition 7.2. This assumption implies that we must know its mean vector and covariance matrix. If we have no information favoring one predicted class over the other, the mean may be considered zero in coordinate space, corresponding to a probability of $1/2$ for each of the possible labels (Fig. 7.2), such that the prior distribution can be written

$$p(\boldsymbol{\pi}, \pi_*|\mathbf{X}, x_*) = \mathcal{N}_{(0,1)^{n+1}}\left(\mathbf{0}, \mathbf{C}\right), \quad \mathbf{C} = \begin{pmatrix} \boldsymbol{\Sigma}(\mathbf{X}) & \boldsymbol{\sigma}(\mathbf{X}, x_*) \\ \boldsymbol{\sigma}(\mathbf{X}, x_*)^T & \sigma_*^2 \end{pmatrix}. \tag{7.9}$$

The several covariances $\boldsymbol{\Sigma}(\mathbf{X})$ among sampled locations and $\boldsymbol{\sigma}(\mathbf{X}, x_*)$ between a sampled location and the unsampled one, are derived from a second-order stationary covariance function, giving smoothness to the hidden random function in feature space. Note that, as a result of the derivations later on, the covariance function enters the final prediction at x_* only through the prior. Hence, as the multiplication of c_0 by a constant can be compensated by adapting **C** according to Remark 7.3, c_0 can be set to 1 in the prior and thus later on in Eq. (7.11).

7.4.2 Doubly Stochastic Gaussian Quadrature

7.4.2.1 Predictive Estimation

As stated above, we are interested in the predictive probability $p(y_*|\mathbf{X}, \mathbf{y}, x_*)$. Following the definition of predictive estimation, we know that

$$\begin{aligned} p(y_*|\mathbf{X}, \mathbf{y}, x_*) &= \int p(y_*, \pi_*|\mathbf{X}, \mathbf{y}, x_*) d\pi_* \\ &= \int p(y_*|\pi_*) p(\pi_*|\mathbf{X}, \mathbf{y}, x_*) d\pi_* \end{aligned}$$

We go on with the calculations as in [149] and [100], using the conditional independence assumptions reflected by Fig. 7.1. This gives

$$
\begin{aligned}
&= \int p(y_*|\pi_*) \int p(\pi_*, \boldsymbol{\pi}|\mathbf{X}, \mathbf{y}, x_*) d\boldsymbol{\pi}\, d\pi_* \\
&= \int p(y_*|\pi_*) \int p(\pi_*|\boldsymbol{\pi}, \mathbf{X}, x_*) p(\boldsymbol{\pi}|\mathbf{X}, \mathbf{y}) d\boldsymbol{\pi}\, d\pi_* \\
&= \frac{1}{c_1} \int p(y_*|\pi_*) \int p(\pi_*|\boldsymbol{\pi}, \mathbf{X}, x_*) p(\mathbf{y}|\boldsymbol{\pi}, \mathbf{X}) p(\boldsymbol{\pi}|\mathbf{X}) d\boldsymbol{\pi}\, d\pi_* \qquad (7.10)
\end{aligned}
$$

Using the probability density function of the normal distribution in $(0,1)^n$, plugging in the coordinate representation (7.8) and repeatedly applying the substitution rule of integration, we obtain explicit expressions for all terms in (7.10), viz.

$$
\begin{aligned}
p(y_*|\pi_*) &= \left(\frac{e^{c_0\phi_*}}{1+e^{c_0\phi_*}}\right)^{y_*} \left(\frac{1}{1+e^{c_0\phi_*}}\right)^{1-y_*} = \frac{e^{c_0\phi_* y_*}}{1+e^{c_0\phi_*}} \\
p(\pi_*|\boldsymbol{\pi}, \mathbf{X}, x_*) &= \frac{1}{\sqrt{\tau\left(\sigma_*^2 - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}\right)}} \exp\left(-\frac{1}{2}\frac{\left(\phi_* - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}\right)^2}{\sigma_*^2 - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}}\right) \qquad (7.11) \\
p(\mathbf{y}|\boldsymbol{\pi}, \mathbf{X}) = p(\mathbf{y}|\boldsymbol{\pi}) &= \prod_{i=1}^{n} \left[\left(\frac{e^{c_0\phi_i}}{1+e^{c_0\phi_i}}\right)^{y_i} \left(\frac{1}{1+e^{c_0\phi_i}}\right)^{1-y_i}\right] = \prod_{i=1}^{n} \frac{e^{c_0\phi_i y_i}}{1+e^{c_0\phi_i}} \\
p(\boldsymbol{\pi}|\mathbf{X}) &= \frac{1}{\sqrt{\tau^n|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}\right)
\end{aligned}
$$

where we have used among others the derivation in [149, chap. 2.2] for Eq. (7.11). In these equations, we have used the shorter notation $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\mathbf{X})$, as well as $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\mathbf{X}, x_*)$.

The integration in (7.10) now is with respect to $\boldsymbol{\phi}$ and ϕ_*, logistic coordinates of the unobservable probabilities $\boldsymbol{\pi}$ and π_*. Inserting the terms mentioned before, we obtain

$$
\begin{aligned}
&p(y_*|\mathbf{X}, \mathbf{y}, x_*) \\
&= c_2 \iint \frac{e^{c_0\phi_* y_*}}{1+e^{c_0\phi_*}} \prod_{i=1}^{n} \frac{e^{c_0\phi_i y_i}}{1+e^{c_0\phi_i}} \exp\left(-\frac{\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}}{2} - \frac{(\phi_* - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi})^2}{2s_*^2}\right) d\boldsymbol{\phi}\, d\phi_* \\
&\qquad (7.12)
\end{aligned}
$$

with $s_*^2 := \sigma_*^2 - \boldsymbol{\sigma}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}$ and $c_2 := (c_1\sqrt{\tau^n|\boldsymbol{\Sigma}|}\sqrt{\tau s_*^2})^{-1}$. This integral cannot be solved in closed form.

7.4.2.2 Approximating the Integral

In this section, we derive a computational scheme for the calculation of the predictive doubly stochastic mode for Gaussian process classification presented before.

Since the integral in Eq. (7.12) cannot be solved analytically, we approximate the exact logistic function in (7.8) by a stretched error function, i.e.

$$\frac{e^{c_0\phi_* y_*}}{1+e^{c_0\phi_*}} \approx \Phi\left((-1)^{y_*+1}k_0\phi_*\right) \tag{7.13}$$

where Φ denotes the error function. This allows for substantial simplifications leading to the result in equation (7.15). Choosing

$$k_0 = \arg\min_k \max_{\phi_*} \left| \frac{e^{c_0\phi_*}}{1+e^{c_0\phi_*}} - \Phi(k\phi_*) \right| \approx 0.5876c_0$$

we obtain a good approximation with a maximum deviation of

$$\max_{\phi_*} \left| \frac{e^{c_0\phi_*}}{1+e^{c_0\phi_*}} - \Phi(k_0\phi_*) \right| < 0.01$$

for every c_0 (see Fig. 7.3). Of course, the same calculation is valid for ϕ_i and $y_i, i = 1, \ldots, n$, instead of ϕ_* and y_*.

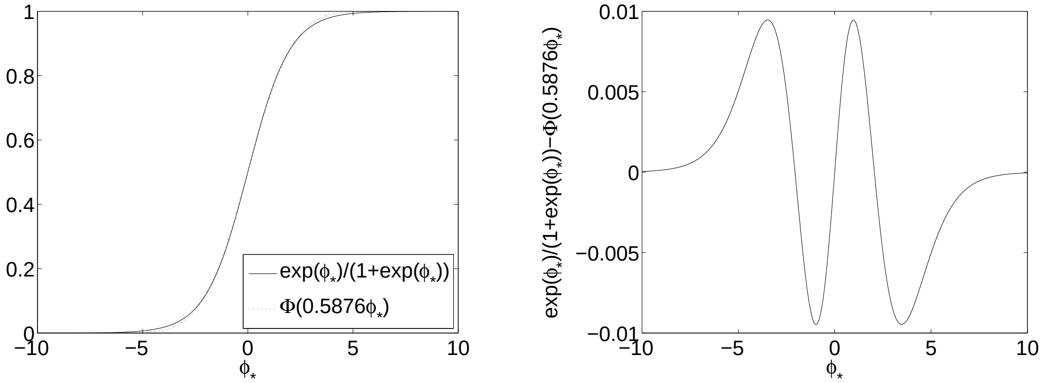


Figure 7.3: Comparison of the original logistic function and its stretched inverse probit approximation for $c_0 = 1$. The left panel shows the two functions, the right panel their difference.

Working toward the final simplification, we define a multivariate generalization of the Heaviside function $H(\xi)$.

Definition 7.4. *Let*

$$\boldsymbol{H}_{\boldsymbol{y}}(\boldsymbol{\xi}) := \begin{cases} 0 & \textit{if} \;\; \exists i : (-1)^{y_i+1}\xi_i < 0 \\ \frac{1}{2} & \textit{if} \;\; \forall i : (-1)^{y_i+1}\xi_i \geq 0 \textit{ and } \exists i : \xi_i = 0 \\ 1 & \textit{if} \;\; \forall i : (-1)^{y_i+1}\xi_i > 0 \end{cases}$$

Special cases of $\mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})$ in one dimension are $\mathbf{H}_1(\xi) = H(\xi)$ and $\mathbf{H}_0(\xi) = H(-\xi) = 1 - H(\xi)$. Summarized in words, the function $\mathbf{H}_{\mathbf{y}}$ is—up to a null set with respect to the Lebesgue measure—equal to 1 in exactly one orthant of $\mathbb{R}^n$ and equal to 0 elsewhere, where the orthant is specified by the components of $\mathbf{y}$.

One can verify that $\Phi(k_0\phi_*) = (\mathbf{H}_1 * \mathcal{N}_{0,\frac{1}{k_0^2}})(\phi_*)$ and $\Phi(-k_0\phi_*) = (\mathbf{H}_0 * \mathcal{N}_{0,\frac{1}{k_0^2}})(\phi_*)$, and hence

$$\prod_{i=1}^{n} \Phi\left((-1)^{y_i+1}k_0\phi_i\right) = \prod_{i=1}^{n} \left(\mathbf{H}_{y_i} * \mathcal{N}_{0,\frac{1}{k_0^2}}\right)(\phi_i) = \left(\mathbf{H}_{\mathbf{y}} * \mathcal{N}_{\mathbf{0},\frac{1}{k_0^2}\mathbf{I}}\right)(\boldsymbol{\phi}). \tag{7.14}$$

Inserting the approximation in Eq. (7.13), Definition 7.4 and Eq. (7.14) in Eq. (7.12), we can continue the main calculation so that

$$\begin{aligned} p(y_*|\mathbf{X},\mathbf{y},x_*) \approx c_2 \int & \left(\mathbf{H}_{\mathbf{y}} * \mathcal{N}_{\mathbf{0},\frac{1}{k_0^2}\mathbf{I}}\right)(\boldsymbol{\phi}) \; \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}\right) \\ \times \int & \left(\mathbf{H}_{y_*} * \mathcal{N}_{0,\frac{1}{k_0^2}}\right)(\phi_*) \; \exp\left(-\frac{1}{2s_*^2}(\phi_* - \boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi})^2\right) d\phi_* \, d\boldsymbol{\phi} \end{aligned}$$

Considering only the inner integral we have

$$\begin{aligned} & \int \left(\mathbf{H}_{y_*} * \mathcal{N}_{0,\frac{1}{k_0^2}}\right)(\phi_*) \; \exp\left(-\frac{1}{2s_*^2}(\phi_* - \boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi})^2\right) d\phi_* \\ =& \iint \mathbf{H}_{y_*}(\xi_*)\mathcal{N}_{0,\frac{1}{k_0^2}}(\phi_* - x_B)d\xi_* \; \exp\left(-\frac{1}{2s_*^2}(\phi_* - \boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi})^2\right) d\phi_* \\ =& \sqrt{\tau s_*^2} \int \mathbf{H}_{y_*}(\xi_*) \underbrace{\int \mathcal{N}_{0,\frac{1}{k_0^2}}(x_B - \phi_*)\mathcal{N}_{\boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi},s_*^2}(\phi_*)d\phi_*}_{\mathcal{N}_{\boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi},s_*^2+\frac{1}{k_0^2}}(\xi_*)} \, d\xi_* \end{aligned}$$

which leads to

$$\begin{aligned}
p(y_*|\mathbf{X},\mathbf{y},x_*) \approx\, & c_3 \iint \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathcal{N}_{0,\frac{1}{k_0^2}\mathbf{I}}(\boldsymbol{\phi}-\boldsymbol{\xi})d\boldsymbol{\xi} \\
& \times \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}\right)\int \mathbf{H}_{y_*}(\xi_*)\mathcal{N}_{\boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi},s_*^2+\frac{1}{k_0^2}}(\xi_*)d\xi_*\,d\boldsymbol{\phi} \\
=\, & c_3 \iint \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{y_*}(\xi_*)\int \mathcal{N}_{0,\frac{1}{k_0^2}\mathbf{I}}(\boldsymbol{\phi}-\boldsymbol{\xi}) \\
& \times \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}\right)\mathcal{N}_{0,s_*^2+\frac{1}{k_0^2}}(\boldsymbol{\sigma}(x_*)^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}-\xi_*)d\boldsymbol{\phi}\,d\boldsymbol{\xi}\,d\xi_*
\end{aligned}$$

Defining $\mathbf{s} := \boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}(x_*)$ and $v := k_0^2/(s_*^2k_0^2+1)$, we obtain (up to a constant multiplier) for the integrand of the inner integral

$$\begin{aligned}
& \exp\left(-\frac{1}{2}\left(\boldsymbol{\phi}-\boldsymbol{\xi}\right)^T k_0^2\left(\boldsymbol{\phi}-\boldsymbol{\xi}\right)-\frac{1}{2}\boldsymbol{\phi}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\phi}-\frac{1}{2}\left(\mathbf{s}^T\boldsymbol{\phi}-\xi_*\right)v\left(\mathbf{s}^T\boldsymbol{\phi}-\xi_*\right)\right) \\
=\, & \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T\underbrace{\left(k_0^2\mathbf{I}+\boldsymbol{\Sigma}^{-1}+v\mathbf{s}\mathbf{s}^T\right)}_{:=\mathbf{R}}\boldsymbol{\phi}+\boldsymbol{\phi}^T\underbrace{\left(k_0^2\boldsymbol{\xi}+v\mathbf{s}\xi_*\right)}_{:=\mathbf{m}}-\frac{1}{2}\boldsymbol{\xi}^Tk_0^2\boldsymbol{\xi}-\frac{1}{2}v\xi_*^2\right) \\
=\, & \exp\left(-\frac{1}{2}(\boldsymbol{\phi}-\mathbf{R}^{-1}\mathbf{m})^T\mathbf{R}(\boldsymbol{\phi}-\mathbf{R}^{-1}\mathbf{m})\right)\exp\left(\frac{1}{2}\mathbf{m}^T\mathbf{R}^{-1}\mathbf{m}-\frac{1}{2}\boldsymbol{\xi}^Tk_0^2\boldsymbol{\xi}-\frac{1}{2}v\xi_*^2\right)
\end{aligned}$$

The second factor is independent of $\boldsymbol{\phi}$ and the first factor is a Gaussian kernel function which integrates to a constant with respect to $\boldsymbol{\phi}$. Combining this constant with c_3 we obtain

$$p(y_*|\mathbf{X},\mathbf{y},x_*) \approx c_4 \iint \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{y_*}(\xi_*)\exp\left(\frac{1}{2}\mathbf{m}^T\mathbf{R}^{-1}\mathbf{m}-\frac{1}{2}\boldsymbol{\xi}^Tk_0^2\boldsymbol{\xi}-\frac{1}{2}v\xi_*^2\right)d\boldsymbol{\xi}\,d\xi_*$$

When resubstituting $\mathbf{m}$ and reordering, the exponent becomes

$$-\frac{1}{2}\boldsymbol{\xi}^T(k_0^2\mathbf{I}-k_0^4\mathbf{R}^{-1})\boldsymbol{\xi}+\boldsymbol{\xi}^Tk_0^2\mathbf{R}^{-1}v\mathbf{s}\xi_*-\frac{1}{2}\xi_*\left(v-v^2\mathbf{s}^T\mathbf{R}^{-1}\mathbf{s}\right)\xi_*$$

with $\mathbf{I}$ the identity matrix. This finally yields our principal result

$$p(y_*|\mathbf{X},\mathbf{y},x_*) \approx c_4 \int \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{y_*}(\xi_*)\exp\left(-\frac{1}{2}\tilde{\boldsymbol{\xi}}^T\boldsymbol{\Lambda}\tilde{\boldsymbol{\xi}}\right)d\tilde{\boldsymbol{\xi}} \tag{7.15}$$

where

$$\tilde{\boldsymbol{\xi}} := (\boldsymbol{\xi},\xi_*) \quad \text{and} \quad \boldsymbol{\Lambda} = \begin{pmatrix} k_0^2\mathbf{I}-k_0^4\mathbf{R}^{-1} & -k_0^2v\mathbf{R}^{-1}\mathbf{s} \\ -k_0^2v\mathbf{s}^T\mathbf{R}^{-1} & v-v^2\mathbf{s}^T\mathbf{R}^{-1}\mathbf{s} \end{pmatrix}$$

The expression says that, to make a prediction under the doubly stochastic model, it suffices to compute the mass of an $(n+1)$-dimensional Gaussian distribution (centered at the origin and with precision matrix $\mathbf{\Lambda}$) in a given orthant. This is illustrated in Fig. 7.4. The covariance structure of the distribution is mainly given by the covariance matrix $\mathbf{\Sigma}$ and the vector $\sigma(x_*)$, i.e. by the relative position of the training points and the test point x_* in feature space. Moreover, the covariance structure also depends on the parameter k_0 which trades off prior and observed evidence. The orthant that is integrated over is picked by the observed training set labels (and setting $y_* = 0$ or $y_* = 1$). The normalizing constant c_4 can be determined by calculating not only the mass in the relevant but also in the adjacent orthant $\{(\boldsymbol{\xi}, \xi_*) \in \mathbb{R}^{n+1} : \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{1-y_*}(\xi_*) = 1\}$ and then using the sum constraint $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) + p(Y = 0|\mathbf{X}, \mathbf{y}, x_*) = 1$. In the left panel of Fig. 7.4, $\sigma(x_*)$ is relatively large and $\mathbf{y} = 0$. Hence, the posterior class prediction for class 0 is relatively large; here, $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) = 0.128$. In the right panel, $\sigma(x_*) = 0$. Consequently, the label of the training point does not influence the posterior prediction at x_* and hence, $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) = 0.5$.

For the actual computation of the integral of the Gaussian density, one can evaluate the multivariate error function at the origin after having adequately mirrored the normal distribution. The multivariate error function is e.g. implemented in R and Matlab based on methods presented in [68].

Remark. For DSGQ, there is a close relationship between the sill parameter (see e.g. [37, chap. 2.2]), which affects the assumed covariance structure and therefore the computation of $\mathbf{\Sigma}$, and the parameter c_0.[4] As already mentioned in Remark 7.3, they together influence the variance of the prior in Eq. (7.9) and therefore govern the tradeoff between prior and evidence for the final prediction. The smaller c_0 and the smaller the sill, the higher the weight of the prior.

7.5 Comparison of the Presented Algorithms

7.5.1 Data

The several methods summarized or presented in this contribution will be illustrated and compared using a typical diagnostic problem: *given a static "image" of a system, can we decide whether it corresponds to a particular (dynamic) regime?* In this particular case, we want to use a map of significant wave height, provided by a numerical forecasting model of the Western Mediterranean Sea, to decide whether that is a *Llevant* storm (a storm with dominating winds from the East) or not. The data (including the plot in

[4] Recall that the corresponding parameter k_0 of the DSGQ simply is proportional to c_0.

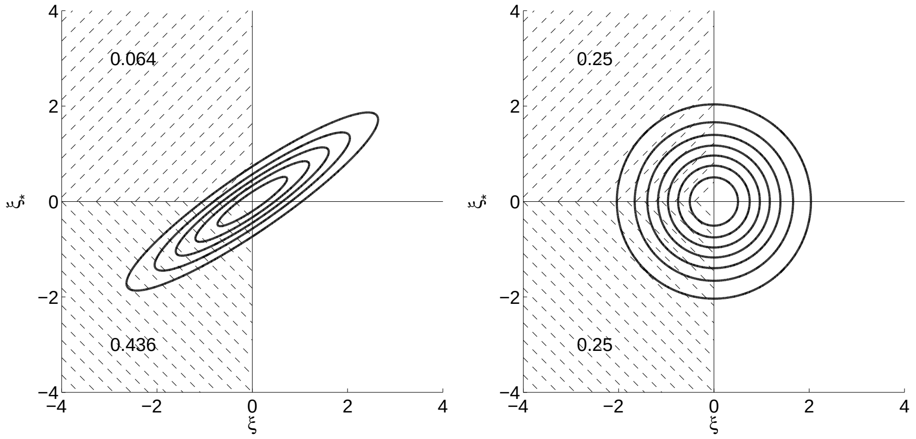


Figure 7.4: Computation of the posterior class probability $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*)$ with the doubly stochastic Gaussian quadrature according to Eq. (7.15). Each panel shows the contour lines of the probability density function of an $(n + 1)$-dimensional Gaussian distribution with 0 mean, where $\xi_* \in \mathbb{R}$ and $\boldsymbol{\xi} \in \mathbb{R}^n$ (obviously, $n = 1$ here). The covariance structure of the distribution mainly reflects the relative positions of the test and training points in feature space. The ratio $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*)/p(Y = 0|\mathbf{X}, \mathbf{y}, x_*)$ equals the ratio of integrals of the Gaussian density over two adjacent orthants, which are determined by the labels **y** of the training points. Here, the regions that are integrated over correspond to $\mathbf{y} = 0$, and $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*)/p(Y = 0|\mathbf{X}, \mathbf{y}, x_*) = 0.064/0.436$ and $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*)/p(Y = 0|\mathbf{X}, \mathbf{y}, x_*) = 0.25/0.25$ in the left and the right panel, respectively. Additionally using the sum constraint $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) + p(Y = 0|\mathbf{X}, \mathbf{y}, x_*) = 1$ yields $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) = 0.128$ and $p(Y = 1|\mathbf{X}, \mathbf{y}, x_*) = 0.5$, respectively.

Fig. 7.5) has been provided by Raimon Tolosana-Delgado from the Maritime Engineering Laboratory (LIM) at the Universitat Politécnica de Catalunya (UPC) in Barcelona.

We have available a set of $n = 114$ such images of past forecasts, for which we now know the dynamic situation. We manually select beforehand a small subset of $d = 8$ "informative" pixels. Subsampling of pixels is performed to avoid the "curse of dimensionality". Otherwise there would be $n = 114$ points in a space with several thousand dimensions d of which many correspond to uninformative locations in the East. As the empirical distribution of the individual wave heights is extremely skewed to the right, they are preprocessed by computing the logarithm. Then, we build a data set of feature vectors $x_i \in \mathbb{R}^8, i = 1, \ldots, n$ (the logarithm of the wave heights at the selected pixel positions) and labels y_i (1 corresponds to a Llevant storm, 0 to "no Llevant storm") and apply the classification techniques to this set. Fig. 7.5 shows the variance of logarithms of significant wave height for the whole forecasting area, using a larger set of 970 non-classified images. We do not consider all these images (but only 114) for the comparison to ensure a high degree of stochastic independence between the images, i.e., we selected the images in such a way that they are at least one week apart from

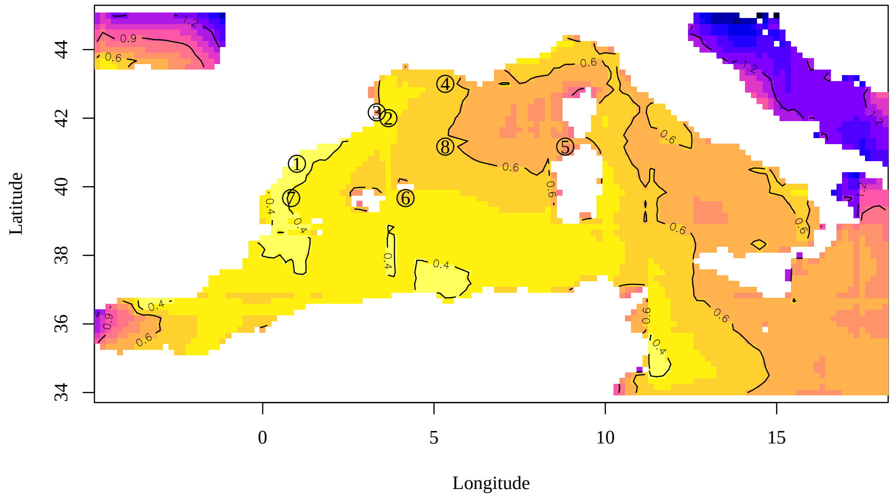


Figure 7.5: The Western Mediterranean with indication of the 8 explanatory features used to classify the forecast images. The contour map shows the variance along each possible feature, i.e. the variance of the logarithm of the wave heights at each pixel. Pixels are 16×16 km^2 approximately.

each other. This figure also shows the locations of the 8 pixels chosen in this case as classification features. Note that the chosen features have moderate, fairly similar variances. Though this is not a necessary condition, it allows us to consider an isotropic variogram on $\mathbb{R}^8$ for the latent Gaussian process.

7.5.2 Experimental Results

We compare the three methods—classical indicator kriging (CIK), simplicial indicator kriging (SIK) and the doubly stochastic Gaussian quadrature (DSGQ)—based on the data presented in the previous section. Throughout this section, we use a Matérn covariance function (see e.g. [149, chap. 4.2]) for all methods and all experiments. Its one-dimensional correlogram is given by

$$\rho(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu r}}{l} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu r}}{l} \right), \quad \nu, l > 0,$$

where K_ν is the modified Bessel function of the second kind [2, chap. 9.6], ν is called a smoothness parameter and l a range parameter. Then, for a given nugget $s_0 > 0$ and a sill $s > s_0$, the covariance function is $h(r) = (s - s_0)\rho(r) + s_0 \mathbf{1}_0(r)$, where $\mathbf{1}_0(\cdot)$ is the indicator function at 0. Hence, $\boldsymbol{\Sigma}_{ij} = h(\|x_i - x_j\|)$ and $\sigma_i = h(\|x_i - x_*\|)$.

In the first experiment, we simply evaluate the classification performance of the 8-

Method	Accuracy	Computation time
Classical indicator kriging	0.868 ± 0.062	0.68 s
Simplicial indicator kriging	0.868 ± 0.062	0.88 s
Doubly stochastic Gaussian quadrature	0.895 ± 0.056	481.18 s

Table 7.1: Relative accuracy and computation time of the three different methods for the classification of the 8-dimensional data. Results are obtained by 5-fold cross validation over 114 samples, "±" indicates the boundaries of the 95% interval. As the parameter estimation is performed differently across the methods, it is not considered for the computation time. The latter is measured with a Matlab implementation run a standard PC.

dimensional data using 5-fold cross-validation (CV): the data is divided into 5 folds; then, 4 of these are used for training to predict the posterior class probabilities for the samples in the remaining fold (test fold). This is repeated 5 times such that each sample is once in the test fold.

For both CIK and SIK the parameters are determined by standard variogram methods [37] yielding a smoothness of $\nu = 10$, a range of $l = 0.4$, a sill of $s = 0.17$, and no nugget effect ($s_0 = 0$). For DSGQ, the function values of the underlying process are not observable, because the class labels are modeled as realizations of Bernoulli experiments. Hence, standard variogram methods are not applicable and we use *nested* CV for parameter estimation. In order to predict posterior probabilities for a test fold in the outer CV, only the data in the respective training folds are used for parameter tuning. This is performed in an inner CV loop. Hence, for different test folds of the outer CV, different parameters may be used. Note that, in contrast to a simple (non-nested) CV, this does not yield overoptimistic estimates for classifier performance as the parameters for predicting probabilities for a test fold in the outer CV loop are tuned completely without using any information about this test fold [178]. For computational reasons, we use the same values for ν, s and s_0 as in the other two methods and optimize l and k_0 only.

The quality indicators of the methods are presented in Table 7.1. On the one hand, the highest accuracy is achieved by the doubly stochastic method (DSGQ).[5] On the other hand, the running time of the DSGQ is much higher, more than 500 the time needed for IK techniques.

Next, in order to get more insight into the differences of the methods, we perform an experiment using only two dimensions of the 8-dimensional data. By visual inspection, we select the second and the third feature as these seem to be the most informative features for classification. The 2-dimensional data is plotted in all panels of Fig. 7.6. We use all samples for training and predict posterior class probabilities on a

[5] However, note that the differences are not statistically significant.

two-dimensional grid. We obtain $\nu = 0.5$, $l = 0.4$, $s = 0.17$ and $s_0 = 0$ for CIK and SIK using variogram methods. For DSGQ, we again use the same values for ν, s and s_0 as in CIK and SIK and optimize the remaining parameters with cross-validation using only the training samples. This leads to $l = 3$ and $k_0 = 4$. The resulting contour plots for all methods are shown in Fig. 7.6.

It can be observed that all points of the *training set* are classified correctly with CIK and SIK, in particular the dissenting points that are located in between a cloud of points with a different label. Here, as the variogram estimate yields $s_0 = 0$, the estimate for the posterior class probabilities at those points are p or $1 - p$ for SIK and even 1 or 0 for CIK. This prevents the assignment of opposite classes in the neighborhood of observed labels and thus limits the generalization ability of CIK and SIK. In contrast, in the doubly stochastic model, the dissenting points are considered unlikely realizations of a Bernoulli experiment. This explains why the squashed realization of the Gaussian process is much smoother for DSGQ, as can be inferred from the contour lines. Hence, the doubly stochastic model is more robust with respect to these dissenting points, even under $s_0 = 0$.

7.6 Conclusions

We have presented a new method for the estimation of the class probabilities in a classification setting, based on a doubly stochastic process formalism. Seen from a Bayesian perspective, the method is obtained as the predictive probability of a prior random field updated by a Bernoulli likelihood obtained from the training set. The distinctive characteristic of the method is that the underlying estimation is deterministic and analytical up to a final step of iterative maximization or integration. The underlying doubly stochastic model is consistent with a classification framework.

In contrast, (classical) indicator kriging (CIK) [90] is theoretically inconsistent, as it uses a non-transformed Gaussian random field (with range $-\infty$ to $+\infty$) to describe a probability (bounded between 0 and 1). SIK uses a logistic-transformed Gaussian RF as reference to avoid negative probabilities. However, both CIK and SIK are interpolators, and thus do not reflect a two-step stochastic process. In particular, this becomes apparent in the presence of conflicting observations (successes surrounded by failures, or vice versa): the posterior probabilities estimated by CIK or SIK can only be exactly 0 or 1 at locations where there are observations. These methods hence categorically rule out the possibility to observe the opposite label at those locations. This is not realistic for typical prediction settings which are characterized by some class overlap. In contrast, DSGQ can take observations from the neighborhood into account and produces more plausible predictions at the site of observations.

Although the accuracy of DSGQ is higher than that of CIK and SIK for the 8-

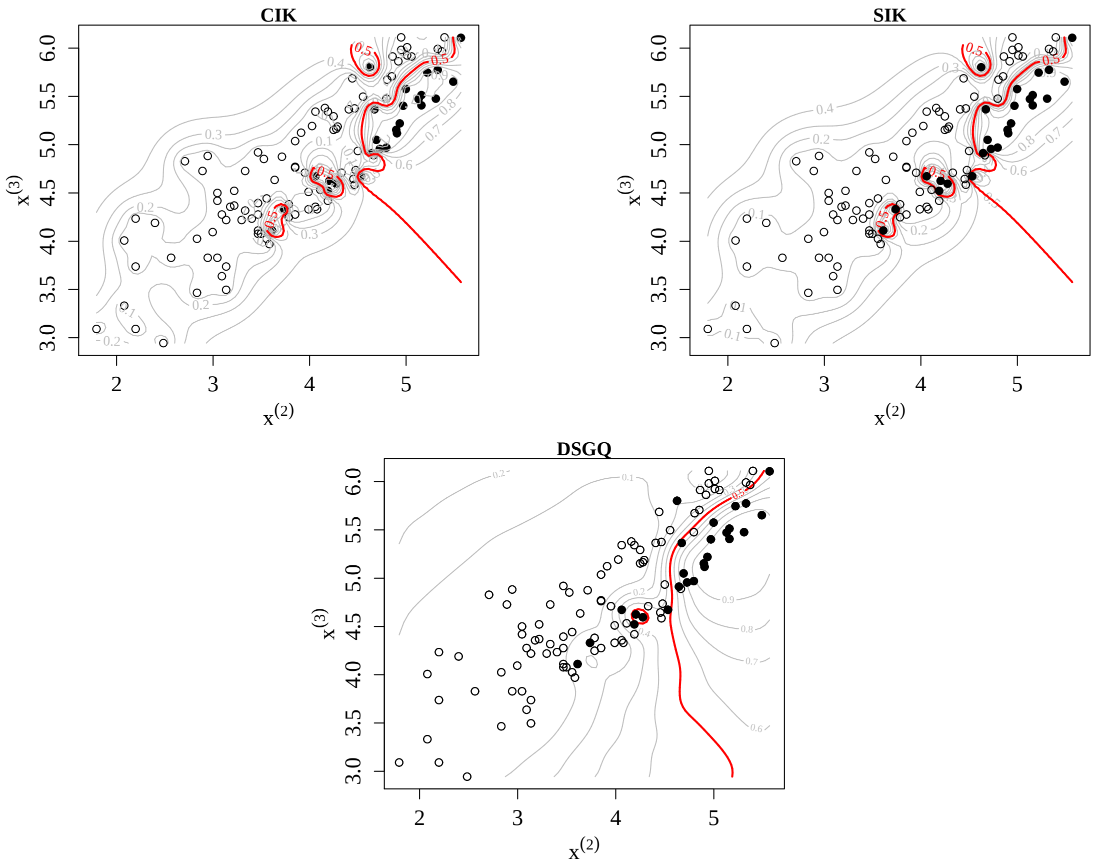


Figure 7.6: Contour plots of the posterior predictions for the four methods compared, based on a two-dimensional projection of the data on the $x^{(2)}$-$x^{(3)}$-plane, i.e. using these two features only. The decision boundary between the two classes, the level curve for $\{x_* : p(Y = 0|\mathbf{X}, \mathbf{y}, x_*) = 0.5\}$, is depicted with a thicker line. Samples of class 0 and 1 are represented by empty and filled circles, respectively. Note that the decision boundary (in contrast to the other contour lines) is equal for CIK and SIK. Both CIK and SIK make predictions that are compatible with each and every label from the training set which is prone to overfitting. In contrast, DSGQ takes dissenting points into account, but do not follow them unconditionally in its predictions.

dimensional data used here, the difference is not significant. The fact that the doubly stochastic model is computationally more demanding than CIK and SIK without showing convincingly better performance is probably the reason why CIK, the classical approach in geostatistics for classification, still is very popular despite its inconsistency. Moreover, all parameters in the underlying statistical model of CIK can easily be interpreted in physical terms.

Note that in [152][6], the three methods discussed here are also compared to the doubly stochastic “Aitchison Maximum Posterior”. The results are similar to those for DSGQ both with respect to performance and computation time.

While the experiments show that the new computational scheme of the DSGQ works in principle, an alternative to the numerical integration is desirable because it may be too expensive or too inexact if n is large. For this, note that we only need to know the ratio of the probability and the complementary probability of obtaining label y_* at the point x_* to make a prediction:

$$\frac{c_4 \int_{\mathbb{R}^{n+1}} \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{y_*}(\xi_*) e^{-\frac{1}{2}\tilde{\boldsymbol{\xi}}^T \Lambda \tilde{\boldsymbol{\xi}}} d\tilde{\boldsymbol{\xi}}}{c_4 \int_{\mathbb{R}^{n+1}} \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{1-y_*}(\xi_*) e^{-\frac{1}{2}\tilde{\boldsymbol{\xi}}^T \Lambda \tilde{\boldsymbol{\xi}}} d\tilde{\boldsymbol{\xi}}} = \frac{\int_{\Omega_1} e^{-\frac{1}{2}(G\tilde{\boldsymbol{\xi}})^T(G\tilde{\boldsymbol{\xi}})} d\tilde{\boldsymbol{\xi}}}{\int_{\Omega_2} e^{-\frac{1}{2}(G\tilde{\boldsymbol{\xi}})^T(G\tilde{\boldsymbol{\xi}})} d\tilde{\boldsymbol{\xi}}} = \frac{\int_{G(\Omega_1)} e^{-\frac{1}{2}\tilde{\boldsymbol{\xi}}'^T \tilde{\boldsymbol{\xi}}'} d\tilde{\boldsymbol{\xi}}'}{\int_{G(\Omega_2)} e^{-\frac{1}{2}\tilde{\boldsymbol{\xi}}'^T \tilde{\boldsymbol{\xi}}'} d\tilde{\boldsymbol{\xi}}'} \tag{7.16}$$

where we have defined $\Omega_1 = \{\tilde{\boldsymbol{\xi}} : \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{y_*}(\xi_*) = 1\}$, $\Omega_2 = \{\tilde{\boldsymbol{\xi}} : \mathbf{H}_{\mathbf{y}}(\boldsymbol{\xi})\mathbf{H}_{1-y_*}(\xi_*) = 1\}$ and have used the Cholesky decomposition $\Lambda = G^T G$ and the multidimensional substitution rule for integration. The regions $G(\Omega_i)$, over which we integrate, are convex cones with apices in the origin (because they are linear transformations of orthants) and the integrand $\exp(-\frac{1}{2}\tilde{\boldsymbol{\xi}}'^T \tilde{\boldsymbol{\xi}}')$ is a radially symmetric function. Hence, the value of the whole integral is proportional to the volume of the intersection of the cone with the unit sphere (called a spherical simplex). Thus, in order to evaluate the fraction (7.16), we need to compute the ratio of the volumes of the spherical simplices determined by $G(\Omega_1)$ and $G(\Omega_2)$ [7]. Finding a tractable approximation to this ratio is an attractive avenue for future research.

[6] This is a self-citation. The “Aitchison Maximum Posterior” is not presented in this chapter because it has been contributed by Raimon Tolosana-Delgado.

8 Conclusions

The main contributions presented in this thesis are the derivation of

- two new approaches to active learning (Chapters 4 and 6),
- distributional estimates for ε-nearest neighbors, k-nearest neighbors, random forests (Chapter 3) and kernel density classification (Chapters 4, used for the first AL strategy at the same place),
- an outlier detection for random forests (Chapter 5, used for the second AL strategy in Chapter 6), and
- a new computational scheme for Gaussian process classification (Chapter 7).

The first AL strategy, DEAL, performs best relative to other strategies if the underlying classifier provides a distributional estimate of the sampling distribution at each unlabeled point. This distributional estimate encodes both the distance to the current decision boundary and the number of labeled samples in the neighborhood of a point. Additionally taking density information into account, a natural definition of the training utility value has been derived leading to a novel active learning strategy. Kernel density classification has been used as the underlying classifier for the implementation of the strategy. The corresponding distributional estimates have been derived in this context. The empirical performance of the AL strategy has been evaluated on a wide range of different data sets. It outperforms random, uncertainty and Look-Ahead Selective Sampling on a wide range of data sets.

Distributional estimates have also been derived for ε-nearest neighbors, k-nearest neighbors and random forests. It has been shown using road sign recognition, IMS and toy data that the distributional estimates, in particular those for random forests, combine state of the art classification performance with the ability of detecting test samples that are not well represented by the training set. Unfortunately, these estimates mainly allow for a relative comparison of posterior estimation uncertainty. The obvious open problem is the derivation of a second-order distribution that indeed approximates the true sampling distribution. Then, random forests could immediately serve as the underlying classifier for DEAL.

Instead, a similar second active learning strategy has been developed to combine the state of the art classification performance of random forests with the main ideas of the first strategy. This strategy has then been applied within the field of automatic optical inspection. The approach has been evaluated on the publicly available DAGM data set. It has been shown empirically that active learning techniques indeed can reduce the labeling effort in industrial quality control.

As a part of the strategy, an outlier detection algorithm based on random forests has been proposed. It is faster and performs significantly better than a method based on similarity matrices that has previously been proposed for random forests. Moreover, the performance of the proposed method is similar to a standard nearest neighbor outlier detection scheme.

Finally, we have derived a new computational scheme for the doubly stochastic model in Gaussian process classification. The method is analytical up to a final step involving numerical integration in a space with dimension equal to the number of training samples plus 1. In order to apply the method for large training sets and to accelerate the estimation, an analytical solution to the final integral is desirable. Note that the method cannot be combined with any of the AL approaches presented in this thesis since only posterior class probabilities are estimated.

List of Abbreviations

AL	Active Learning
AUC	Area Under the (ROC-) Curve
CIK	Classical Indicator Kriging
CRF	Confidence Random Forest(s)
CV	Cross Validation
DEAL	Distributional Estimate Active Learning
DSGQ	Doubly Stochastic Gaussian Quadrature
IMS	Imaging Mass Spectrometry
NN	Nearest Neighbors
oob	out of bag
RBF	Radial Basis Function
RF	Random Forest(s)
ROC	Receiver Operating Characteristic
SIK	Simplicial Indicator Kriging
SSL	Semi-Supervised Learning
TUV	Training Utility Value

List of Figures

List of Tables

Bibliography

[1] Abe, N and Mamitsuka, H (1998): Query Learning Strategies Using Boosting and Bagging. In: *Proceedings of the International Conference on Machine Learning (ICML)*, 1–9.

[2] Abramowitz, M and Stegun, IA (1965): *Handbook of Mathematical Functions*. Dover, New York.

[3] Acciani, G and Fornarelli, G (2006): Application of Neural Networks in Optical Inspection and Classification of Solder Joints in Surface Mount Technology. *IEEE Transactions on Industrial Informatics* **2**(3), 200–209.

[4] Agresti, A and Coull, BA (1998): Approximate is Better than “Exact” for Interval Estimation of Binomial Proportions. *The American Statistician* **52**(2), 119–126.

[5] Andrews, S; Tsochantaridis, I and Hofmann, T (2003): Support Vector Machines for Multiple-Instance Learning. In: *Advances in Neural Information Processing Systems (NIPS)*, 561–568.

[6] Angluin, D (1988): Queries and Concept Learning. *Machine Learning* **2**, 319–342.

[7] Aomoto, K (1977): Analytic Structure of the Schläfli Function. *Nagoya Mathematical Journal* **68**, 1–16.

[8] Argamon-Engelson, S and Dagan, I (1999): Committee-Based Sample Selection for Probabilistic Classifiers. *Journal of Artificial Intelligence Research* **11**, 335–360.

[9] Frank, A and Asuncion, A (2010): UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`. Irvine, CA: University of California, School of Information and Computer Science.

[10] Balcan, M; Beygelzimer, A and Langford, J (2009): Agnostic Active Learning. *Journal of Computer and System Sciences* **75**(1), 78–89.

[11] Banfield, RE; Hall, LO; Bowyer, KW and Kegelmeyer, WP (2007): A Comparison of Decision Tree Ensemble Creation Techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(1), 173–180.

[12] Baram, Y; El-Yaniv R and Luz, K (2004): Online Choice of Active Learning Algorithms. *Journal of Machine Learning Research* **5**, 255–291.

[13] Barnett, V and Lewis, T (1994): *Outliers in Statistical Data*. John Wiley & Sons, 3rd edition.

[14] Baum, EB and Lang, K (1992): Query Learning Can Work Poorly When a Human Oracle Is Used. In: *Proceedings of the IEEE International Joint Conference on Neural Networks.*

[15] Begleiter, R; El-Yaniv, R and Pechyony, D (2008): Repairing Self-Confident Active-Transductive Learners Using Systematic Exploration. *Pattern Recognition Letters* **29**, 1245–1251.

[16] Ben-Gal, I (2005): Outlier detection. In: Maimon O and Rockach L (Eds.): *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers.* Kluwer Academic Publishers.

[17] Biau, G and Devroye, L (2010): On the Layered Nearest Neighbour Estimate, the Bagged Nearest Neighbour Estimate and the Random Forest Method in Regression and Classification. *Journal of Multivariate Analysis* **101**(10), 2499–2518.

[18] Biau, G; Devroye, L and Lugosi, G (2008): Consistency of Random Forests and Other Averaging Classifiers. *Journal of Machine Learning Research* **9**, 2015–2033.

[19] Billheimer, D; Guttorp, P and Fagan, WF (2001): Statistical Interpretation of Species Composition. *Journal of the American Statistical Association* **96**, 1205–1214.

[20] Bishop, CM (2006): *Pattern Recognition and Machine Learning.* Springer, New York.

[21] Bonwell, CC and Eison, JA (1991): *Active Learning: Creating Excitement in the Classroom (AEHE-ERIC Higher Education Report No.1).* Jossey-Bass, Washington D.C.

[22] Box, GEP; Hunter, JS and Hunter, WG (2005): *Statistics for Experimenters: Design, Innovation, and Discovery.* John Wiley & Sons, 2nd edition.

[23] Bradley, AP (1997): The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern Recognition* **30**(7), 1145–1159.

[24] Breiman, L (2001): Random Forests. *Machine Learning* **45**(1), 5–32.

[25] Breiman, L (2003): Manual – Setting Up, Using and Understanding Random Forests V4.0. Technical Report, UC Berkeley. Available at http://oz.berkeley.edu/users/breiman/Using_random_forests_v4.0.pdf.

[26] Breiman, L (2004): Consistency for a Simple Model of Random Forests. Technical Report 670, Statistics Department, UC Berkeley.

[27] Breiman, L; Friedman, JH; Olshen, RA and Stone, CJ (1984): *CART: Classification and Regression Trees.* Wadsworth, Belmont.

[28] Brodley, CE and Friedl, MA (1999): Identifying Mislabeled Training Data. *Journal of Artificial Intelligence Research* **11**, 131–167.

[29] Brown, LD; Cai, TT and DasGupta, A (2001): Interval Estimation for a Binomial Proportion. *Statistical Science* **16**(2), 101–133.

[30] Carr, J and Mao, N (1993): A General Form of Probability Kriging for Estimation of the Indicator and Uniform Transforms. *Mathematical Geology* **25**(4), 425–438.

[31] Cebron, N and Berthold, MR (2009): Active Learning for Object Classification: from Exploration to Exploitation. *Data Mining and Knowledge Discovery* **18**, 283–299.

[32] Chandola, V; Banerjee, A and Kumar, V (2009): Anomaly Detection: A Survey. *ACM Computing Surveys* **41**(3), Article 15.

[33] Chapelle, O; Zien, A and Schölkopf, B (Eds.) (2006): *Semi-Supervised Learning*. MIT Press.

[34] Chatzichristofis, S and Boutalis, Y (2008): CEDD: Color and Edge Directivity Descriptor: A Compact Descriptor for Image Indexing and Retrieval. In: *Proceedings of the 6th international conference on Computer Vision Systems*, 312–322.

[35] Chen, C; Liaw, A and Breiman, L (2004): Using Random Forest to Learn Unbalanced Data. Technical Report 666, Department of Statistics, University of California. Available at http://www.stat.berkeley.edu/users/chenchao/666.pdf.

[36] Cheng, HD; Shan, J; Ju, W; Guo, Y and Zhang, L (2010): Automated Breast Cancer Detection and Classification Using Ultrasound Images: A Survey. *Pattern Recognition* **43**(1), 299–317.

[37] Chilès, JP and Delfiner, P (1999): *Geostatistics: Modeling Spatial Uncertainty*. John Wiley & Sons, New York.

[38] Chow, CK (1970): On Optimum Recognition Error and Reject Tradeoff. *IEEE Transactions on Information Theory* **16**(1), 41–46.

[39] Cohn, DA; Ghahramani, Z and Jordan, MI (1996): Active Learning with Statistical Models. *Journal of Artificial Intelligence Research* **4**, 129–145.

[40] Culotta, A and McCallum, A (2005): Reducing Labeling Effort for Structured Prediction Tasks. In: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 746–751.

[41] Currin, C; Mitchell, T; Morris, M and Ylvisaker, D (1991): Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments. *Journal of the American Statistical Association* **86**(416), 953–963.

[42] Curuana, R and Niculescu-Mizil, A (2006): An Empirical Comparison of Supervised Learning Algorithms. In: *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 161–168.

[43] Dagan, I and Engelson, S (1995): Committee-Based Sampling for Training Probabilistic Classifiers. In: *Proceedings of the International Conference on Machine Learning (ICML)*, 150–157.

[44] Dasarathy, BV (1980): Nosing Around the Neighborhood: A New System Structure and Classification Rule for Recognition in Partially Exposed Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2**(1), 67–71.

[45] Demšar, J (2006): Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research* **7**, 1–30.

[46] Devarakota, PRR; Mirbach B and Ottersten, B (2008): Reliability Estimation of a Statistical Classifier. *Pattern Recognition Letters* **29**, 243–253.

[47] Devroye, L; Györfi, L; Krzyżak, A and Lugosi, G (1994): On the Strong Universal Consistency of Nearest Neighbor Regression Function Estimates. *The Annals of Statistics* **22**(3), 1371–1385.

[48] Devroye, L; Györfi, L and Lugosi, G (1996): *A Probabilistic Theory of Pattern Recognition*. Springer, New York.

[49] Devroye, L and Krzyżak, A (1989): An Equivalence Theorem for L_1 Convergence of the Kernel Regression Estimate. *Journal of Statistical Planning and Inference* **23**, 71–82.

[50] Dietterich, TG; Lathrop, RH and Lozano-Perez, T (1997): Solving the Multiple-Instance Problem with Axis-Parallel Rectangles. *Artificial Intelligence* **89**, 31–71.

[51] Diggle, PJ; Tawn, JA and Moyeed, RA (1998): Model-Based Geostatistics (with Discussion). *Journal of the Royal Statistical Society, Series C (Applied Statistics)* **47**(3), 299–350.

[52] Dubuisson, B and Masson, M (1993): A Statistical Decision Rule with Incomplete Knowledge about Classes. *Pattern Recognition* **26**(1), 155–165.

[53] Duda, R; Hart, P and Stork, D (2001): *Pattern Classification*. Wiley-Interscience.

[54] Engelson, SP and Dagan, I (1996): Minimizing manual annotation cost in supervised training from corpora. In: *Proceedings of the 34th Annual Meeting*, Association for Computational Linguistics, 319–326.

[55] Ertekin, Ş; Huang, J; Bottou, L and Giles, CL (2007): Learning on the Border: Active Learning in Imbalanced Data Classification. In: *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, 127–136.

[56] Escalante, HJ (2005): A Comparison of Outlier Detection Algorithms for Machine Learning. In: *Proceedings of the International Conference on Communications in Computing.*

[57] Everitt, BS; Landau, S and Leese, Morven (2009): *Cluster analysis*. John Wiley & Sons, 4th edition.

[58] Faber, NM (2003): Sample-Specific Standard Error of Prediction for Partial Least Squares Regression. *Trends in Analytical Chemistry*, **22**(5), 330–334.

[59] Fawcett, T (2006): An Introduction to ROC Analysis. *Pattern Recognition Letters* **27**, 861–874.

[60] Fergus, R; Perona, P and Zisserman, A (2003): Object class recognition by unsupervised scale-invariant learning. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 264–271.

[61] Ferreira, MJ; Santos, C and Monteiro, J (2009): Cork Parquet Quality Control Vision System Based on Texture Segmentation and Fuzzy Grammar. *IEEE Trabsactions on Industrial Electronics* **56**(3), 756–765.

[62] Freund, Y; Seung, HS; Shamir, E and Tishby, N (1997): Selective Sampling Using the Query by Committee Algorithm. *Machine Learning* **28**, 133–168.

[63] Friedman, J.H. (1997): On Bias, Variance, 0/1 Loss, and the Curse of Dimensionality, *Data Mining and Knowledge Discovery* **1**(1), 55–77.

[64] Fujii, A; Tokunaga, T; Inui, K and Tanaka, H (1998): Selective Sampling for Example-Based Word Sense Disambiguation. *Computational Linguistics* **24**(4), 573–597.

[65] Gan, G; Ma, C and Wu, J (2007): *Data Clustering: Theory, Algorithms, and Applications*. SIAM, Society for Industrial and Applied Mathematics.

[66] Gao, D; Zhang, Y and Zhao, Y (2009): Random Forest Algorithm for Classification of Multiwavelength Data. *Research in Astronomy and Astrophysics* **9**(2), 220–226.

[67] García-Magariños, M; López-de-Ullibarri, I; Cao, R and Salas, A (2009): Evaluating the Ability of Tree-Based Methods and Logistic Regression for the Detection of SNP-SNP Interaction. *Annals of Human Genetics* **73**, 360–369.

[68] Genz, A and Bretz, F (2009): *Computation of Multivariate Normal and t Probabilities*. Lecture Notes in Statistics, Vol. 195. Springer, Heidelberg.

[69] Geurts, P; Fillet, M; de Seny, D; Meuwis, M; Malaise, M; Merville, M and Wehenkel, L (2005): Proteomic Mass Spectra Classification Using Decision Tree Based Ensemble Methods. *Bioinformatics* **21**(15), 3138–3145.

[70] Gibbs, MN (1997): *Bayesian Gaussian Processes for Classification and Regression*. PhD Thesis, University of Cambridge.

[71] Gibbs, MN and MacKay, DJC (2000): Variational Gaussian Process Classifiers. *IEEE Transactions on Neural Networks* **11**(6), 1458–1464.

[72] Gneiting, T and Raftery, AE (2007): Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, **102**(477), 359–378.

[73] Görlitz, L (2007): *Modern Concepts for Semi-Supervised Learning and Multidimensional Image Processing*. University of Heidelberg, PhD Thesis.

[74] Grall-Maës, E and Beauseroy, P (2009): Optimal Decision Rule with Class-Selective Rejection and Performance Constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31**(11), 2073–2082.

[75] Guo, Y and Schuurmans, D (2008): Discriminative Batch Mode Active Learning. In: *Advances in Neural Information Processing Systems (NIPS)*, 593–600.

[76] Ha, TM (1997): The Optimum Class-Selective Rejection Rule. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(6), 608–615.

[77] Hampel, FR; Ronchetti, EM; Rousseeuw, PJ and Stahel, WA (1986): *Robust Statistics: The Approach Based on Influence Functions.* John Wiley & Sons, New York.

[78] Hand, DJ (2006): Classifier Technology and the Illusion of Progress. *Statistical Science* **21**(1), 1–14.

[79] Hanselmann, M; Köthe, U; Kirchner, M; Renard, BY; Amstalden, ER; Glunde, K; Heeren, RMA and Hamprecht, FA (2009): Toward Digital Staining using Imaging Mass Spectrometry and Random Forests. *Journal of Proteome Research* **8**(7), 3558–3567.

[80] Härdle, W; Müller, M; Sperlich, S and Werwatz, A (2004): *Nonparametric and Semiparametric Models.* Springer, Heidelberg.

[81] Hastie, T; Tibshirani, R and Friedman, J (2009): *The Elements of Statistical Learning. Data Mining, Inference and Prediction.* Springer, New York, 2nd edition.

[82] Hawkins, D (1980): *Identification of Outliers.* Chapman and Hall.

[83] Hodge, VJ and Austin, J (2004): A Survey of Outlier Detection Methodologies. *Artificial Intelligence Review* **22**, 85–126.

[84] Hoi, S; Jin, R; Zhu, J and Lyu, M (2006): Batch Mode Active Learning and its Application to Medical Image Classification. In: *Proceedings of the 23rd International Conference on Machine Learning*, 417–424.

[85] Huber, PJ (2004): *Robust Statistics: Theory and Methods.* John Wiley & Sons, New York.

[86] Hwa, R (2004): Sample Selection for Statistical Parsing. *Computational Linguistics* **30**(3), 73–77.

[87] Hwang, W; Runger, G and Tuv, E (2007): Multivariate Statistical Process Control with Artificial Contrasts. *IIE Transactions* **39**, 659–669.

[88] Jeffreys, H (1946): An Invariant Form for the Prior Probability in Estimation Problems. *Proceedings of the Royal Society A* **186**, 453–461.

[89] Joshi, AJ; Porikli, F and Papanikolopoulos, N (2009): Multi-class Active Learning for Image Classification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2372–2379.

[90] Journel, AG (1983): Nonparametric Estimation of Spatial Distributions. *Mathematical Geology* **15**(3), 445–468.

[91] Journel, AG and Posa, D (1990): Characteristic Behavior and Order Relations for Indicator Variograms. *Mathematical Geology* **22**(8), 1011–1025.

[92] Jurafsky, D and Martin, JH (2008): *Speech and Language Processing*. Prentice Hall, 2nd edition.

[93] Kapoor, A; Grauman, K; Urtasun, R and Darrell, T (2007): Active Learning with Gaussian Processes for Object Categorization. In: *International Conference on Computer Vision (ICCV)*, 1–8.

[94] Karras, DA; Karkanis, SA; Iakovidis, D; Maroulis, DE and Mertzios, BG (2001): Support Vector Machines for Improved Defect Detection in Manufacturing Using Novel Multidimensional Wavelet Feature Extraction Involving Vector Quantization and PCA Techniques. In: *NIMIA Advanced Study Institute on Neural Networks for Instrumentation, Measurement and Related Industrial Applications*, 139–144.

[95] Kaufman, L and Rousseeuw, PJ (2005): *Finding Groups in Data: An Introduction to Cluster Analysis* (Wiley Series in Probability and Statistics). Wiley-Interscience.

[96] Kazianka, H and Pilz, J (2010): Copula-Based Geostatistical Modeling of Continuous and Discrete Data Including Covariates. *Stochastic Environmental Research and Risk Assessment* **24**(5), 661–673.

[97] Kim, SC and Kang, TJ (2006): Automated Defect Detection System Using Wavelet Packet Frame and Gaussian Mixture Model. *Journal of the Optical Society of America* **23**(11), 2690–2701.

[98] Kononenko, I (2001): Machine Learning for Medical Diagnosis: History, State of the Art and Perspective. *Artificial Intelligence in Medicine* **23**, 89–109.

[99] Kumar, A (2008): Computer-Vision-Based Fabric Defect Detection: A Survey. *IEEE Transactions on Industrial Electronics* **55**(1), 348–363.

[100] Kuss, M and Rasmussen, CE (2005): Assessing Approximate Inference for Binary Gaussian Process Classification. *Journal of Machine Learning Research* **6**, 1679–1704.

[101] Landgrebe, TCW; Tax, DMJ; Paclík, P and Duin, RPW (2006): The Interaction between Classification and Reject Performance for Distance-Based Reject-Option Classifiers. *Pattern Recognition Letters* **27**, 908–917.

[102] LeCun, Y; Matan, O;Boser, B; Denker, JS; Henderson, D; Howard, RE; Hubbard, W; Jackel, LD and Baird, HS (1990): Handwritten Zip Code Recognition with Multilayer Networks. In: *Proceedings of the International Conference on Pattern Recognition*, II:35–40.

[103] Lee, HS; Kim, HI; Cho, NI; Jeong, YH; Chung, KS and Jun, CS (2005): Automatic Defect Classification Using Boosting. In: *Proceedings of the Fourth International Conference on Machine Learning and Applications (ICMLA)*, 357–362.

[104] Lewis, D and Catlett, J (1994): Heterogeneous Uncertainty Sampling for Supervised Learning. In: *Proceedings of the International Conference on Machine Learning (ICML)*, 148–156.

[105] Lewis, D and Gale, W (1994): A Sequential Algorithm for Training Text Classifiers. In: *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 3–12.

[106] Li, M and Sethi, IK (2006): Confidence-Based Active Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(8), 1251–1261.

[107] Liaw, A and Wiener, M (2002): Classification and Regression by RandomForest. *R News: The Newsletter of the R Project* **2**(3), 18–22. Available at http://cran.r-project.org/doc/Rnews/.

[108] Lim, YB; Sacks, J; Studdeen, W and Welch, W (2002): Design and Analysis of Computer Experiments when the Output is Highly Correlated over the Input Space. *The Canadian Journal of Statistics* **30**(1), 109–126.

[109] Lin, H (2009): Automated Defect Inspection of Light-Emitting Diode Chips Using Neural Network and Statistical Approaches. *Expert Systems with Applications* **36**, 219–226.

[110] Lin, Y and Jeon, Y (2006): Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association* **101**, 578–590.

[111] Lindenbaum, M; Markovitch, S and Rusakov, D (2004): Selective Sampling for Nearest Neighbor Classifiers. *Machine Learning* **54**(2), 125–152.

[112] Linder, M and Sundberg, R (2002): Precision of Prediction in Second-Order Calibration, with Focus on Bilinear Regression Models, *Journal of Chemometrics*, **16**, 12–27.

[113] Liu, J and MacGregor, J (2006): Estimation and Monitoring of Product Aesthetics: Application to Manufacturing of “Engineered Stone” Countertops. *Machine Vision and Applications* **16**(6), 374–383.

[114] Lomasky, R; Brodley, CE; Aernecke, M; Walt, D and Friedl, M (2007): Active Class Selection. In: *Proceedings of the European Conference on Machine Learning (ECML)*, 640–647.

[115] Lorigo, LM and Govindaraju, V (2006): Offline Arabic Handwriting Recognition: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(5), 712–724.

[116] Lugosi, G and Nobel, A (1996): Consistency of Data-Driven Histogram Methods for Density Estimation and Classification. *The Annals of Statistics* **24**(2), 687–706.

[117] Lunetta, KL; Hayward, LB; Segal, J and and Eerdewegh, PV (2004): Screening Large-Scale Association Study Data: Exploiting Interactions Using Random Forests. *BMC Genetics* **5**, Article Number 32.

[118] Mandriota, C; Nitti, M; Ancona, N; Stella, E and Distante, A (2004): Filter-Based Feature Selection for Rail Defect Detection. *Machine Vision and Applications* **15**, 179–185.

[119] Mantero, P; Moser, G and Serpico, SB (2005): Partially Supervised Classification of Remote Sensing Images Through SVM-Based Probability Density Estimation. *IEEE Transactions on Geoscience and Remote Sensing* **43**(3), 559–570.

[120] Martin, JK (1997): An Exact Probability Metric for Decision Tree Splitting and Stopping. *Machine Learning* **28**, 257–291.

[121] Mathew, T and Kasala, S (1994): An Exact Confidence Region in Multivariate Calibration. *The Annals of Statistics*, **22**(1), 94–105.

[122] McCallum, AK and Nigam, K (1998): Employing EM in Pool-Based Active Learning for Text Classification. In: *Proceedings of the Fifteenth International Conference on Machine Learning (ICML)*, 350–358.

[123] McDonnell, LA and Heeren, RMA (2006): Imaging Mass Spectrometry. *Mass Spectrometry Reviews* **26**, 606–643.

[124] Mehta, CR and Patel, NR (1995): Exact Logistic Regression: Theory and Examples. *Statistics in Medicine* **14**, 2143–2160.

[125] Melville, P and Mooney, R (2004): Diverse Ensembles for Active Learning. In: *Proceedings of the International Conference on Machine Learning (ICML)*, 584–591.

[126] Minka, TP (2001): *A Family of Algorithms for Approximate Bayesian Inference*. PhD Thesis, Massachusetts Institute of Technology.

[127] Mitchell, T (1997): *Machine Learning*. McGraw-Hill.

[128] Mitchell, T (2006): The discipline of machine learning. Technical Report CMU-ML-06-108, Carnegie Mellon University.

[129] Moosmann, F; Nowak, E and Jurie, F (2008): Randomized Clustering Forests for Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **30**(9), 1632–1646.

[130] Mosimann, JE (1962): On the Compound Multinomial Distribution, the Multivariate β-Distribution, and Correlations Among Proportions. *Biometrika* **49**(1/2), 65–82.

[131] Muslea, I; Minton, S and Knoblock, CA (2000): Selective sampling with redundant views. In: *Proceedings of the National Conference on Artificial Intelligence*, 621–626.

[132] Muslea, I; Minton, S and Knoblock, CA (2002): Active + Semi-supervised Learning = Robust Multi-View Learning. In: *Proceedings of the Nineteenth International Conference on Machine Learning*, 435–442.

[133] Neal, RM (1999): Regression and Classification Using Gaussian Process Priors. In: Bernardo, JM; Berger, JO; Dawid, AP and Smith AFM (Eds.): *Bayesian Statistics 6*, Oxford University Press, 475–501.

[134] Nguyen, HT and Smeulders, A (2004): Active Learning Using Pre-Clustering. In: *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 623–630.

[135] Niculescu-Mizil, A and Caruana, R (2005): Predicting Good Probabilities with Supervised Learning. In: *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, 625–632.

[136] Olivieri, AC; Faber, NM; Ferre, J; Boque, R; Kalivas, JH and Mark, H (2006): Uncertainty Estimation and Figures of Merit for Multivariate Calibration. *Pure Applied Chemistry* **78**(3), 633–661.

[137] Opper, M and Winther, O (2000): Gaussian Processes for Classification: Mean Field Algorithms. *Neural Computation* **12**(11), 2655–2684.

[138] Osugi, T; kun, D and Scott, S (2005): Balancing Exploration and Exploitation: A New Algorithm for Active Machine Learning. In: *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM)*, 330–337.

[139] Paalanen, P; Kamarainen, J; Ilonen, J and Kälviäinen, H (2006): Feature Representation and Discrimination Based on Gaussian Mixture Model Probability Densities – Practices and Algorithms. *Pattern Recognition* **39**, 1346–1358.

[140] Pal, M (2005): Random Forest Classifier for Remote Sensing Classification. *International Journal of Remote Sensing* **26**(1), 217–222.

[141] Pardo-Igúzquiza, E and Dowd, P (2005): Multiple Indicator Cokriging with Application to Optimal Sampling for Environmental Monitoring. *Computers & Geosciences* **31**(1), 1–13.

[142] Pawlowsky-Glahn, V (2003): Statistical Modelling on Coordinates. In: Thió-Henestrosa S, Martín-Fernández JA (Eds.): *Compositional Data Analysis Workshop – CoDaWork'03*, Universitat de Girona, http://dugi-doc.udg.edu/handle/10256/648.

[143] Pawlowsky-Glahn, V and Egozcue, JJ (2001): Geometric Approach to Statistical Analysis on the Simplex. *Stochastic Environmental Research and Risk Assessment* **15**, 384–398.

[144] Penny, KI and Jolliffe, IT (2001): A Comparison of Multivariate Outlier Detection Methods for Clinical Laboratory Safety Data. *The Statistician* **50**(3), 295–308.

[145] Pernkopf, F (2004): Detection of Surface Defects on Raw Steel Blocks Using Bayesian Network Classifiers. *Pattern Analysis and Applications* **7**, 333–342.

[146] Peters, S (2008): Automated Design of a Hybrid Texture Analysis System for Defect Detection. In: *Proceedings of the 32nd OAGM/AAPR Workshop*, 19–28.

[147] Provost, F and Fawcett, T (1997): Analysis and Visualization of Classifier Performance: Comparison Under Imprecise Class and Cost Distributions. In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD-97)*, 43–48.

[148] Rasmussen, CE (1996): *Evaluation of Gaussian Processes and Other Methods for Non-Linear Regression*. PhD Thesis, Graduate Department of Computer Science, University of Toronto.

[149] Rasmussen, CE and Williams, CKI (2006): *Gaussian Processes for Machine Learning*. MIT Press, Cambridge.

[150] Ripley, BD (1997): *Pattern Recognition and Neural Networks*. Cambridge University Press.

[151] Röder, J; Nadler, B; Kunzmann, K and Hamprecht, FA (2012): Active Learning with Distributional Estimates. In: *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, 715–725.

[152] Röder, J; Tolosana-Delgado, R and Hamprecht, FA (2011): Gaussian Process Classification: Singly Versus Doubly Stochastic Models, and New Computational Schemes. *Stochastic Environmental Research and Risk Assessment* **25**(7), 865–879.

[153] Roy, N and McCallum, A (2001): Toward Optimal Active Learning through Sampling Estimation of Error Reduction. In: *Proceedings of the International Conference on Machine Learning (ICML)*, 441–448.

[154] Rue, H; Martino, S and Chopin, N (2009): Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)* **71**, 319–392.

[155] Sacks, J; Welch, WJ; Mitchell, TJ and Wynn, HP (1989): Design and Analysis of Computer Experiments. *Statistical Science* **4**(4), 409–435.

[156] Schein, AI and Ungar, LH (2007): Active Learning for Logistic Regression: an Evaluation. *Machine Learning* **68**, 235–265.

[157] Schohn, G and Cohn, D (2000): Less is More: Active learning with Support Vector Machines. In: *Proceedings of the Seventeenth International Conference on Machine Learning*, 839–846.

[158] Schoonard, JW; Gould, JD and Miller, LA (1973): Studies of Visual Inspection. *Ergonomics* **16**(4), 365–379.

[159] Scott, DW (1992): *Multivariate Density Estimation*. John Wiley & Sons, New York.

[160] Seeger, M (2001): *Learning with Labeled and Unlabeled Data*. Technical Report, University of Edinburgh.

[161] Segal, MR (2004): Machine Learning Benchmarks and Random Forest Regression. Technical Report, eScholarship Repository, University of California, `http://repositories.cdlib.org/cbmb/bench_rf_regn/`.

[162] Seligson, DB; Horvath, S; Shi, T; Yu, H; Tze, S; Grunstein, M and Kurdistani, SK (2005): Global Histone Modification Patterns Predict Risk of Prostate Cancer Recurrence. *Nature* **435**, 1262–1266.

[163] Settles, B (2010): Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison.

[164] Settles, B and Craven, M (2008): An Analysis of Active Learning Strategies for Sequence Labeling Tasks. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1069–1078.

[165] Seung, HS; Opper, M and Sompolinsky, H (1992): Query by committee. In: *Proceedings of the ACM Workshop on Computational Learning Theory*, 287–294.

[166] Shi, T and Horvath, S (2006): Unsupervised Learning with Random Forest Predictors. *Journal of Computational and Graphical Statistics* **15**(1), 118–138.

[167] Siegler, RS (1976): Three Aspects of Cognitive Development. *Cognitive Psychology* **8**, 481–520.

[168] Silvén, O; Niskanen, M and Kauppinen, H (2003): Wood Inspection with Non-Supervised Clustering. *Machine Vision and Applications* **13**, 275–285.

[169] Simoncelli, EP and Freeman, WT (1995): The Steerable Pyramid: A Flexible Architecture for Multi-Scale Derivative Computation. In: *Proceedings of the 2nd IEEE Conference on Image Processing*, 444–447.

[170] Steinwart, I; Hush, D and Scovel, C (2005): A Classification Framework for Anomaly Detection. *Journal of Machine Learning Research* **6**, 211–232.

[171] Suro-Perez, V and Journel, AG (1991): Indicator Principal Component Kriging. *Mathematical Geology* **23**(5), 759–788.

[172] Tax, DMJ and Duin, RPW (2001): Uniform Object Generation for Optimizing One-Class Classifiers. *Journal of Machine Learning Research* **2**, 155–173.

[173] Tenenbaum, JB; Griffiths, TL and Kemp, C (2006): Theory-Based Bayesian Models of Inductive Learning and Reasoning. *Trends in Cognitive Sciences* **10**(7), 309–318.

[174] Tibshirani, R (1996): A Comparison of Some Error Estimates for Neural Network Models, *Neural Computation* **8**, 152–163.

[175] Tolosana-Delgado, R; Pawlowsky-Glahn, V and Egozcue, JJ (2008): Indicator kriging Without Order Relation Violations. *Mathematical Geosciences* **40**, 327–347.

[176] Tong, S and Koller, D (2001): Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research* **2**, 45–66.

[177] Tukey, JW (1977): *Exploratory Data Analysis.* Addison-Wesley, Reading, Massachusetts.

[178] Varma, S and Simon, R (2006): Bias in Error Estimation when Using Cross-Validation for Model Selection. *BMC Bioinformatics* **7**, 91.

[179] Vezhnevets, A and Barinova, O (2007): Avoiding Boosting Overfitting by Removing Confusing Samples. In: *Proceedings of the Eighteenth European Conference on Machine Learning (ECML)*, 430–441.

[180] Walstra, A; Sauer, P; Wieler, M; Perwass, C and Hamprecht, FA (2010): A Generic Defect Detection Framework for Fully Automated Industrial Quality Control. Unpublished manuscript.

[181] Wasserman, L (2005): *All of Non-Parameteric Statistics*. Springer.

[182] Wiener, N (1949): *Extrapolation, Interpolation and Smoothing of Stationary Time Series with Engineering Applications.* John Wiley & Sons, New York.

[183] Williams, CKI and Barber, D (1998): Bayesian Classification with Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(12), 1342–1351.

[184] Williams, CKI and Rasmussen, CE (1996): Gaussian Processes for Regression. In: *Advances in Neural Information Processing Systems (NIPS)*, 514–520.

[185] Xie, X (2008): A Review of Recent Advances in Surface Defect Detection Using Texture Analysis Techniques. *Electronic Letters on Computer Vision and Image Analysis* **7**(3), 1–22.

[186] Xie, X and Mirmehdi, M (2007): TEXEMS: Texture Exemplars for Defect Detection on Random Textured Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(8), 1454–1464.

[187] Yu, X (2007): *Prediction Intervals for Class Probabilities*. Master Thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand. Available at `http://researchcommons.waikato.ac.nz/handle/10289/2436`.

[188] Zhang, C and Chen, T (2002): An Active Learning Framework for Content-Based Information Retrieval. *IEEE Transactions on Multimedia* **4**(2), 260–268.

[189] Zhang, J and Zulkernine, M (2006): Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection. In: *Proceedings of the IEEE International Conference on Communications (ICC)-Symposium on Network Security and Information Assurance*, 2388–2393.

[190] Zhou, ZH; Chen, KJ and Dai, HB (2006): Enhancing Relevance Feedback in Image Retrieval Using Unlabeled Data. *ACM Transactions on Information Systems (TOIS)* **24**(2), 219–244.

[191] Zhu, M and Ghodsi, A (2006): Automatic Dimensionality Selection from the Scree Plot Via the Use of Profile Likelihood. *Computational Statistics & Data Analysis* **51**, 918–930.

[192] Zhu, X (2005): *Semi-Supervised Learning Literature Survey*. Computer Sciences Technical Report 1530, University of Wisconsin-Madison.

[193] Zhu, X; Ghahramani, Z and Lafferty, J (2003): Combining Active Learning and Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In: *Proceedings of the ICML Workshop on the Continuum from Labeled to Unlabeled Data*, 58–65.

[194] Zhu, X; Rogers, T; Qian, R and Kalish, C (2007): Humans Perform Semi-Supervised Classification too. In: *Twenty-Second AAAI Conference on Artificial Intelligence (AAAI-07)*, 864–869.

[195] Zhu, X; Wu, X and Chen, Q (2003): Eliminating Class Noise in Large Datasets. In: *Proceedings of the Twentieth International Conference on Machine Learning*, 920–927.

List of Publications

Journals:

- **Röder, J**; Tolosana-Delgado, R and Hamprecht, FA (2011): Gaussian Process Classification: Singly Versus Doubly Stochastic Models, and New Computational Schemes. *Stochastic Environmental Research and Risk Assessment* **25**(7), 865–879.

Peer-reviewed Conferences:

- **Röder, J**; Nadler, B; Kunzmann, K and Hamprecht, FA (2012): Active Learning with Distributional Estimates. In: *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI),* 715–725.

- **Röder, J**; Nadler, B; Hanselmann, M and Hamprecht, FA: Bayesian Distributional Uncertainty Estimates for Local Averaging Classifiers. *Submitted to the International Conference on Machine Learning (ICML).*

Extended Abstracts:

- Hanselmann, M; **Röder, J**; Köthe, U; Renard, BY; Kreshuk, A; Heeren, RMA and Hamprecht, FA (2010): Active Learning for Efficient Labeling and Classification of Imaging Mass Spectrometry Data. *58th ASMS Conference,* Salt Lake City, Utah, USA.