Learning to recognise.
A study on one-class classification and active learning.

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. dr. ir. J.T. Fokkema,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 8 Juni 2006 om 12.30 uur

door

Piotr JUSZCZAK

Magister Inżynier fizyki Politechniki Wrocławskiej,
geboren te Oława, Polen.

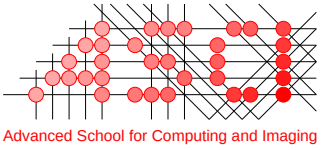
Dit proefschrift is goedgekeurd door de promotoren:
Prof. dr. ir. M.J.T. Reinders

Toegevoegd promotor: Dr. ir. R.P.W Duin

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. M.J.T. Reinders,	Technische Universiteit Delft, promotor
Dr. ir. R.P.W Duin,	Technische Universiteit Delft, toegevoegd promotor
Dr. L.I. Kuncheva,	University of Wales, Bangor
Prof. dr. K. Roos,	Technische Universiteit Delft
Prof. dr. ir. A.W.M. Smeulders,	Universiteit van Amsterdam
Prof. dr. M.H. Overmars,	Universiteit Utrecht
Dr. T.M. Heskes,	Radboud Universiteit Nijmegen
Prof. dr. ir. F.W. Jansen,	Technische Universiteit Delft, reservelid

This work was partly supported by the Dutch Organisation for Scientific Research (NWO).



This work was carried out in graduate school ASCI. ASCI dissertation series number 123.

ISBN-10: 90-9020684-1
ISBN-13: 978-90-9020684-4

Learning to recognise.
A study on one-class classification and active learning.

Thesis

presented for the degree of Doctor of Philosophy
at Delft University of Technology,
under the authority of the Vice-Chancellor, Prof. Dr. Ir. J.T. Fokkema,
to be defended in public in the presence of a committee
appointed by the Board for Doctorates
on 8 June 2006 at 12.30

by

Piotr JUSZCZAK

MSc in physics from Wrocław University of Technology,
born in Oława, Poland.

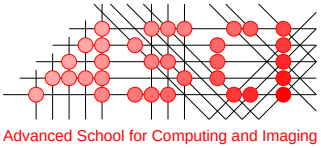
this thesis is approved by the supervisor:
Prof. Dr. Ir. M.J.T. Reinders

Adjunct supervisor: Dr. Ir. R.P.W Duin

Composition of the Doctoral Examination Committee:

Vice-Chancellor,	Chairman
Prof. Dr. Ir. M.J.T. Reinders,	Delft University of Technology, supervisor
Dr. Ir. R.P.W Duin,	Delft University of Technology, adjunct supervisor
Dr. L.I. Kuncheva,	University of Wales, Bangor
Prof. Dr. K. Roos,	Delft University of Technology
Prof. Dr. Ir. A.W.M. Smeulders,	University of Amsterdam
Prof. Dr. M.H. Overmars,	Utrecht University
Dr. T.M. Heskes,	Radboud University Nijmegen
Prof. Dr. Ir. F.W. Jansen,	Delft University of Technology, reserve member

This work was partly supported by the Dutch Organisation for Scientific Research (NWO).



This work was carried out in graduate school ASCI. ASCI dissertation series number 123.

ISBN-10: 90-9020684-1
ISBN-13: 978-90-9020684-4

To my parents

Contents

Notation and basic terminology

$\mathbf{1}$	$N \times 1$ vector of ones $[1, 1 \ldots, 1]^T$
$\mathbf{A}, \mathbf{a}$	centre of N-sphere in input space
$\mathbf{a}_{\mathcal{H}}$	centre of the largest empty sphere, inside SVDD in Hilbert space
$\mathbf{A}_{\mathcal{H}}$	centre of SVDD sphere in Hilbert space
α, β	weights
$\mathbf{c}$	centre of an ellipsoid
$\det(M)$	determinant of a matrix M
$d(\mathbf{x}\|X_t)$	distance from $\mathbf{x}$ to target class
e	edge of a graph
ε_t	target rejection rate
ε_o	outlier acceptance rate
ε_t^{tr}	target rejection rate estimated on a training set
E	$N \times N$ shape matrix of an ellipsoid
$\mathcal{E}_{E,\mathbf{c}}$	ellipsoid with centre $\mathbf{c}$ and shape matrix E
$\mathcal{F}$	active learning function
γ	complexity parameter of a classifier
Γ	gamma function
$\mathcal{H}$	Hilbert space
h	classifier
h_c	height of spherical cap
$\mathcal{I}$	indicator function
r_c	radius of spherical cap
K	kernel matrix
L	Lagrangian, the combination of an error function and constrains
Λ	weighted sum of errors
$\boldsymbol{\mu}$	mean vector of a dataset
n	size of a training set
N	dimensionality of input space
$\mathcal{N}$	normal distribution
m	size of an unlabelled (test) set
$\tilde{M}$	$N+1 \times N+1$ shape matrix of an ellipsoid
ω	class label
$\mathbf{p}(\mathbf{x})$	projection of the vector $\mathbf{x}$
$p(\mathbf{x}\|X_t)$	probability that $\mathbf{x}$ belongs to target class
$P(\mathbf{x})$	probability density function of $\mathbf{x}$
$P(\mathbf{x}\|X_t)$	probability density function of $\mathbf{x}$ given X_t
$r_{\mathcal{H}}$	radius of the largest empty sphere, inside SVDD in Hilbert space
R, r	radius of N-sphere
$R_{\mathcal{H}}$	radius of SVDD sphere in Hilbert space
$\mathbb{R}^N$	input space
$\mathcal{S}(\mathbf{A}, R), \mathcal{S}(\mathbf{a}, r)$	N-sphere with centre $\mathbf{A}/\mathbf{a}$ and radius R/r
Σ	covariance matrix
$\mathbf{sv}$	support vector
θ	threshold on a probability or a distance

$\mathbf{x}$	$N \times 1$ input pattern
ξ	slack variable
X_-	$N \times \binom{N+1}{2}$ matrix of elements $\mathbf{x}_i - \mathbf{x}_j$
X_+	$N \times \binom{N+1}{2}$ matrix of elements $\mathbf{x}_i + \mathbf{x}_j$
X_t	training set
X_u	unlabelled set
V_{cap}	volume of a spherical cap
V_h	volume of a classifier
V_o	volume of a union between two N-spheres
$V_{\mathcal{S}(\mathbf{a},r)}$	volume of N-sphere with a centre $\mathbf{a}$ and radius r
$\mathcal{V}$	$\mathcal{V}$-statistic
$\|\ \|$	Euclidian norm $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$
$\succ, \succeq$	positive definite/semidefinite,
$M \succ 0$	symmetric positive definite matrix. This is equivalent to $\lambda(M) > 0$ or $\mathbf{x}^T M\mathbf{x} > 0,\ \forall \mathbf{x} \neq 0$
$M \succeq 0$	symmetric positive semidefinite matrix. This is equivalent to $\lambda(M) \geq 0$ or $\mathbf{x}^T M\mathbf{x} \geq 0,\ \forall \mathbf{x} \neq 0$

Abbreviations

1-NN	1-Nearest Neighbour
AUC	Area Under ROC Curve
err	estimation of error reduction sampling
EM	Expectation Maximisation
EM-qbc	Expectation Maximisation query by committee
k-NN	k Nearest Neighbour
LDA	Linear Discriminant Analysis
LPDD	Linear Programming Data Description
MST	Minimum Spanning Tree
MST_DD	Minimum Spanning Tree Data Description
MVEB	Minimum Volume Enclosing Box
MVEE	Minimum Volume Enclosing Ellipsoid
MVEE_DD	Minimum Volume Enclosing Ellipsoid Data Description
MVES	Minimum Volume Enclosing Sphere
MDS	Multi Dimensional Scaling
MoG	Mixture of Gaussians
MPM	Maximum Probability Machine
NFLM	Nearest Feature Line Method
ROC	Receiver-Operating Characteristic
SOM	Self Organising Map
SVM	Support Vector Machine
SVDD	Support Vector Data Description
qbb	Query by Bagging
QDA	Quadratic Discriminant Analysis

`PCA`	Principle Components Analysis
`pdc`	Positive Density Correction
`ra`	Random Sampling
`us`	Uncertainty Sampling
`vila`	Variation in Label Assignments
`w-us`	Weighted Uncertainty Sampling

Chapter 1

Introduction

This thesis explores the field of pattern recognition from several perspectives. We will investigate questions arising both in the fields of supervised and semi-supervised learning, dealing with diverse issues such as one-class classification, novelty and outlier detection, model selection and enhancement of the classification models by unlabelled data. All topics covered in the first two parts of the thesis are related to the problem of one-class classification, novelty detection and outlier detection. The third part of the thesis covers such topics as active and semi-supervised learning, where one enhances the classification models by incorporating additional knowledge from unlabelled data.

To set the scene for the thesis we want to pick up the terms *learning* and *recognition* from the title and briefly introduce their meaning in our context. Next, the introduction to the problem of one-class classification is given, followed by the outline of the thesis. All three main parts of the thesis can be read independently after one has read the introduction.

1.1 Learning from examples

Learning is an ability of all living organisms in particular humans. We all know what learning is although we have sometimes difficulties to describe it to others. Our description usually contains words like learn, learnt, learned or learning. We think of learning as referring to two components - a process and an outcome. Learning comes with focusing on a particular subject and filtering, rejecting most of other irrelevant things. Moreover, learning is certainly not a static process, it requires action from a learner, it requires to ask questions.

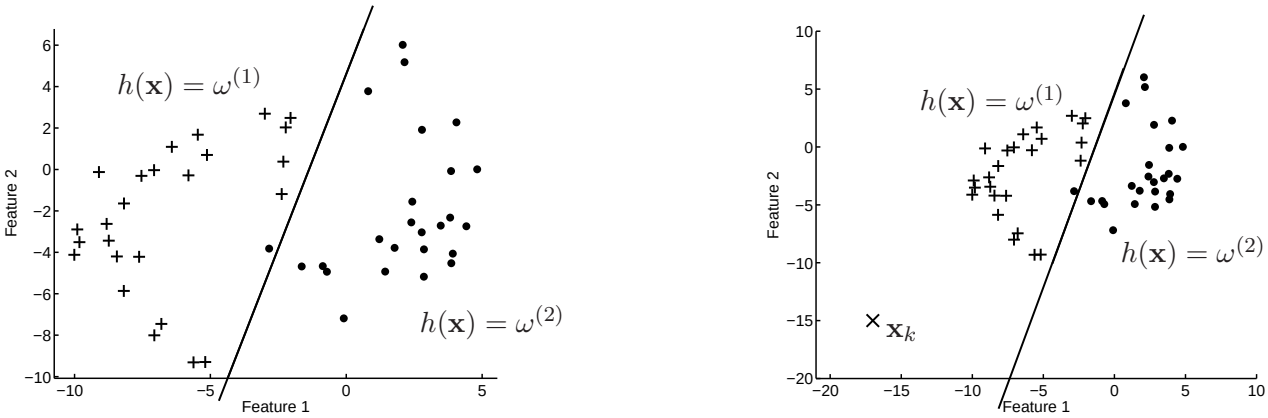
The field of pattern recognition, which is the background of this thesis, tries to imitate the ability of learning. However, traditionally pattern recognition heavily focuses on a classification of two or more classes. In this thesis, we try to get a step closer to the foundations of learning and recognition. In particular, we are going to study the problem of one-class classification where the interest is the recognition of a class of objects and not discrimination between a given set of classes. Moreover, we allow the classifiers to evolve in time by giving them a possibility to ask questions.

Therefore, we use the word *learning* as the process of inferring rules from given examples. The examples are instances of some input space (pattern space), and the rules can consist of some general observation about the structure of the input space, or have the form of a functional dependency between the input and some output space.

To show the difference between recognition and discrimination, in context of pattern recognition, we first compare multi-class classification with one-class classification.

In multi-class classification we are given a set of training objects $X_t := \{(\mathbf{x}_i, \omega_i)|, \mathbf{x}_i \in \mathbb{R}^N, i = 1, \ldots, n\}$, each of the objects consisting of a N dimensional pattern $\mathbf{x}_i$ and its label $\omega_i \in \boldsymbol{\omega}$. Therefore, each object is represented by N features, measurements and can be visualised as a vector in an N dimensional space. The goal is to infer a rule which can assign the correct label ω_k to a new, previously unseen pattern $\mathbf{x}_k$; see figure 1.1(b). In figure 1.1 each object is represented as a vector in $2D$ space spanned by two features.

Further, we assume that for the representation of objects in the input space, $\mathbb{R}^N$, the continuity assumptions holds. This is a general assumption in pattern recognition: two objects near in input space, $\mathbb{R}^N$, should also resemble each other in real life. When we are at one position in the input space, representing an example object, and we change the position slightly, then this new position should represent a very similar object. When this assumption is satisfied for a large part of the input space we only need few examples to learn a decision boundary between classes. However, if the objects from the same class are scattered more or less randomly huge amounts of labelled examples are needed.



(a) (b)

Figure 1.1: (a) A multi-class classifier. (b) A multi-class classifier. $\mathbf{x}_k$ a single outlier object.

A general multi-class classification problem is usually decomposed in several two-class classification problems [Duda et al., 2001]. Therefore, the two class problem is considered the basic classification problem. In a two-class classification problem, the two classes are labelled $\omega^{(1)}$ and $\omega^{(2)}$. For the classification, a function $h(\mathbf{x})$ has to be inferred from the training set. This function should be constructed such that for a given input vector $\mathbf{x}$ an estimate of the label is obtained, $\omega = h(\mathbf{x}|X_t)$:

$$h(\mathbf{x}|X_t) : \mathbb{R}^N \rightarrow \{\omega^{(1)}, \omega^{(2)}\} \tag{1.1}$$

In figure 1.1(a) an example of a two class classification problem is given. The training objects are separated by a linear classifier, plotted as a solid line. A single object from a training set is misclassified. In figure 1.1(b) an additional single test object, $\mathbf{x}_k$, is plotted. The object $\mathbf{x}_k$ is classified to the $\omega^{(1)}$ class. However, we can ask a question if the classifier $h(\mathbf{x}|X_t)$ has sufficient information to classify $\mathbf{x}_k$ to any of the given classes $\omega^{(1)}$ or $\omega^{(2)}$. Any test object on the left side of the decision boundary is classified to $\omega^{(1)}$ and on the right to $\omega^{(2)}$. Even if the distance of a test object to any of the objects from the training set is large. Therefore, any test object, even if it is an outlier is classified to one of the classes represented in a training set.

In such a situation we would like to have a third option: a label which allows a classifier to say *"I do not know"*. This suggests to describe the training set by an enclosing boundary. Only objects that fall inside the boundary are classified to one of the classes represented in the training set [Tax, 2001]; see figure 1.2(a).

An other solution is to enclose individual classes by separate close boundaries 1.2(b). Only objects that fall inside any of the boundaries are classified to one of the classes represented in the training set [Juszczak and Duin, 2003].

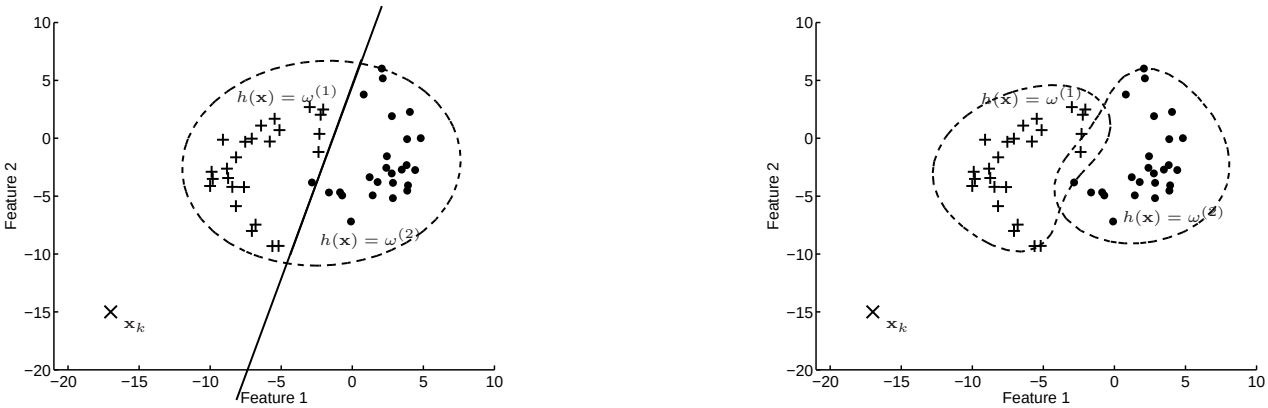


Figure 1.2: (a) A multi-class classifier and a one-class classifier that encloses a training set. (b) A set of one-class classifiers trained on each of the classes separately.

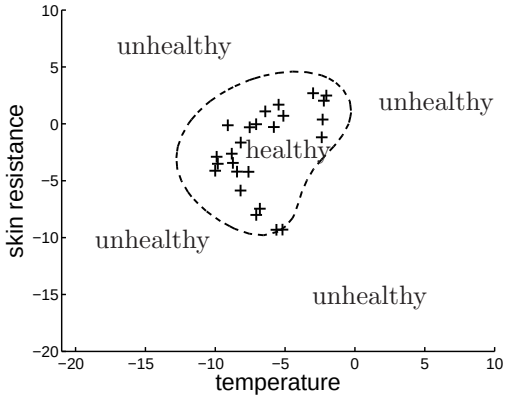
Objects outside the boundaries are classified as outliers, the *"I do not know"* class. The classifiers that focus on description of classes instead of discriminating between classes are called one-class classifiers. The name originated from Moya [Moya et al., 1993].

If we expect that during classification outlier objects, or objects from classes that are not represented in the training set might occur or we only have examples from a single class in the training set, one-class classifiers should be used instead of, or together with multi-class classifiers.

To give the reader some intuition in which classification problems one-class classifiers can be of help we describe the following classification problem.

1.1.1 Practical problem

Imagine that we would like to build a device to classify between the normal and abnormal condition of a person. Assume that, the device is portable and can be worn all the time without discomfort. The decision about a condition of the person is made based on various physiological signals, e.g. `ECG`, blood pressure, skin resistance, temperature, etc. Generally, the condition of a particular person is normally healthy. We do not have (or have very few) examples of abnormal physiological conditions for this particular person. Therefore, based on given examples, we can only build a boundary descriptor, that describes a set of normal conditions of the person; the one-class classifier. The descriptor is trained, in the space of the physiological signals that we are measuring. Figure 1.3 shows a two dimensional measurement space spanned by measurements of temperature and skin resistance. The description of a normal condition, denoted by a dashed line, is trained on collected examples of normal conditions. If measurements are not inside the description they are classified as an abnormal state in person health.



Why can we not train a multi-class classifier? Simply because we have no examples of abnormal physiological states. Even if we had a few of these examples, our task is to detect all abnormal states. We can not ask the person to be sick, moreover with particular symptoms. It is much easer to determine the normal set of physiological states.

Figure 1.3: One-class classifier.

Assume now a different problem. The condition of the person has changed. For example, he has started to exercise and becomes more healthy or maybe had a stroke and becomes more unhealthy. This affects the physiological measurements, however, the condition, as stable, still remains normal for this person. In such cases we need to update the descriptor e.g. by retraining it on examples of the current conditions.
This practical problem raises two questions:

1. Is it possible to train a classifier only on examples from a single class, in such a way that examples of the class are inside the description and all other classes are outside it?

2. How to efficiently improve a classifier for classification problems that change in time?

These two pattern recognition problems are the two main topics of the thesis. The next section gives an introduction to the problem of one-class classification.

1.2 Introduction to one-class classification

In the problem of one-class classification [Moya et al., 1993, Tax, 2001] one of the classes, called the *target class*, has to be distinguished from all other possible objects, which are considered as *outliers*. The need for solving such a task comes from many practical applications. Examples are any type of fault detection [Ypma and Duin, 1998] or target detection, e.g. face detection in images, abnormal behaviour, disease detection [Tarassenko et al., 1995], person identification, authorship verification [Koppel and Schler, 2004], etc.. The methodology for handling such situations, however, can also be useful for imbalanced data cases as one-class classifiers can be trained for each class separately [Juszczak and Duin, 2003]; see figure 1.2(b). The problem of one-class classification is characterised by the presence of a well sampled target class. The goal is to determine a proximity function of an object to the target class such that resembling objects will be accepted as targets and outliers will be rejected. It is assumed that a well-sampled training set of target objects is available, while no (or very few) outlier examples are present. The reason for this assumption is practical, since outliers may occur only occasionally or their measurements might be very costly. Moreover, even when outliers are available in a training stage, they may not always be trusted, as they are badly sampled, with unknown priors and ill-defined distributions. In essence, outliers are weakly defined as they may appear as any kind of deviation or anomaly from the target examples. Still, one-class classifiers need to be trained to optimise the errors on both classes.

In principle, one-class classification methods should refer to all possible knowledge that one has about the target class. The model description of this class should be large enough to accept most new targets, yet sufficiently tight to reject the majority of outliers. This is, however, an ill-posed problem since knowledge about a class is deduced from a finite set of target examples, while the outliers are sampled infrequently or not at all.

Outlier identification is an old topic in statistical data analysis [Barnett and Lewis, 1994], usually approached through *robust statistics*. In general, robust statistics emerged as a family of techniques for estimating the parameters of parametric models while dealing with deviations from idealised assumptions [Rousseeuw and van Driessen, 1999]. It investigates the effects of deviations from modelling assumptions, usually those of normality and the independence of the random errors. Robust parameter estimators are proposed that make use of quantiles, ranks, trimmed means, medians, censoring of particular observations, sample weighting, etc. Deviations include all types of rounding errors due to inaccuracy in data collection, contamination by outliers and departure from assumed sample distributions. Outliers are believed to deviate severely from the characteristics exhibited by the majority of the data, usually due to errors in data collection, copying or computation. So, they are often assumed to be caused by human error. Outliers can also arise from sampling errors, where some members are drawn from a different population than the remaining examples, faulty research methodology, or from faulty distributional assumptions. But they can also be legitimate cases sampled from the correct population. As outliers generally increase the error variance of the parametric methods and can seriously bias or influence statistical estimates, their identification is important. Multivariate methods used for their detection often rely on a robust estimate of the Mahalanobis distance and the comparison with critical values of the χ^2 distribution [Rousseeuw and van Driessen, 1999].

In general, we distinguish two types of outliers. One of these is the set of atypical examples of a target class e.g. due to noise. The other is the set of non-target class examples.

1.2.1 One-class classifiers

One-class classifiers are trained to accept target examples and reject outliers. The basic assumption about an object belonging to a class is that it is similar to other examples within this class. Let $X_t = \{\mathbf{x}_i |\ \mathbf{x}_i \in \mathbb{R}^N, i = 1, \ldots, n\}$ be a training set drawn from the target distribution $p(\mathbf{x})$. Assume a characterisation of this target class by a one-class classifier is sought. In general, one-class classifiers can be presented in the following form:

$$h(\mathbf{x}|X_t, \gamma) = \mathcal{I}(p(\mathbf{x}|X_t, \gamma) > \theta) = \begin{cases} 1, \mathbf{x} \text{ is a target,} \\ 0, \mathbf{x} \text{ is an outlier,} \end{cases} \tag{1.2a}$$

or

$$h(\mathbf{x}|X_t, \gamma) = \mathcal{I}(d(\mathbf{x}|X_t, \gamma) < \theta) = \begin{cases} 1, \mathbf{x} \text{ is a target,} \\ 0, \mathbf{x} \text{ is an outlier,} \end{cases} \tag{1.2b}$$

where the function h models the similarity, in equation (1.2a), and the distance in equation (1.2b), of a vector $\mathbf{x}$ to the training, target data X_t. θ is a specified threshold and $\mathcal{I}(\cdot)$ is the indicator function. Furthermore, γ indicates the complexity of the model. The threshold θ is optimised to reject a certain, usually user-specified, fraction of the target class ε_t^{tr}.
ε_t^{tr} has to be determined by the user for a given application. For example, one can specify the maximum number of allowed false alarms in a machine condition monitoring. Apart from the

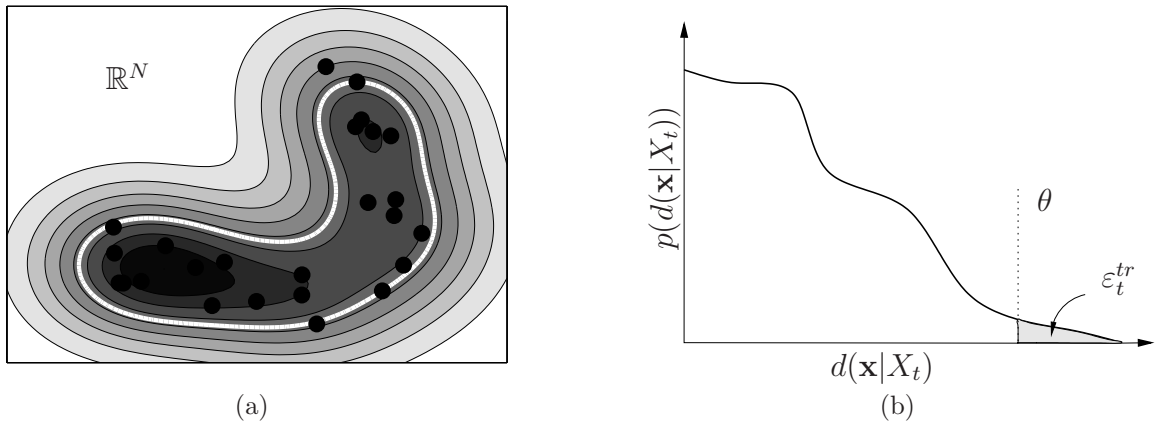


Figure 1.4: (a) A one-class classifier with different thresholds denoted by isolines of different grey values. (b) One dimensional distribution of $d(\mathbf{x}|X_t, \gamma)$. Objects from the target class are distributed according to $p(d(\mathbf{x}|X_t))$.

threshold θ, the performance of a one-class classifier is determined by the complexity parameter γ. It would be possible to determine a complexity parameter γ during training when both errors on the target and outlier class could be estimated e.g. using cross-validation.

Given a fixed target acceptance rate, $1 - \varepsilon_t^{tr}$, the threshold θ is derived from the training set such that the one-class classifier accepts $1 - \varepsilon_t^{tr}$ of the target class, see figure 1.4. That is, given n training samples, θ is determined such that:

$$\frac{1}{n}\sum_{i=1}^{n} \mathcal{I}\{(d(\mathbf{x}|X_t) \geq \theta\} = \varepsilon_t^{tr}. \tag{1.3}$$

where $d(\mathbf{x}|X_t)$ is estimated on the training set. To avoid overfitting, a better estimate might be obtained by using an additional validation set, if one has sufficient amount of data.

1.2.2 Error estimation for one-class classifiers

In general the error of a one-class classifier can be expressed as:

$$\Lambda(\varepsilon_t, \varepsilon_o) = \lambda\varepsilon_t + (1 - \lambda)\varepsilon_o, \tag{1.4}$$

where λ is a trade-off parameter, ε_o denotes the outlier acceptance rate and ε_t target rejection rate. If $\lambda = 0.5$ both errors are equally treated. However, because in one-class classification problems, during training, only examples of the target class are available only ε_t can be reliably estimated. The expected error on the outlier class ε_o can only be estimated by making additional assumptions. Having given estimates of ε_t and ε_o the complexity of a classifier γ and a threshold θ can be optimised.

As only the probability $p(\omega_t|\mathbf{x})$ is known, the error on the target class, the target rejection rate ε_t can be minimised. The outlier acceptance rate, ε_o can only be estimated when example of outliers are used or when additional assumptions about their distribution are made. Note that those two errors are also called the errors of the first and second kind. Table 1.1 shows all possible situations of classifying an object in one-class classification problems.

Table 1.1: Four situations of classifying an object in one-class classification. The false negative ε_o and false positive ε_t correspond to objects which are wrongly classified. Moreover, the true positive $1-\varepsilon_t$ and true negative $1-\varepsilon_o$ correspond to objects which are correctly classified.

		true label	
		target	outlier
estimated label	target	$1-\varepsilon_t$	ε_o
	outlier	ε_t	$1-\varepsilon_o$

Receiver-Operating Characteristic.

To study the behaviour of one-class classifiers, a Receiver-Operating Characteristics (ROC) curve can be used [Bradley, 1997, Tax, 2001], which is a function of the true positive ratio (target acceptance) $(1-\varepsilon_t)$ versus the false positive ratio (outlier acceptance), ε_o. See figure 1.5 for an example. To estimate the outlier acceptance rate outlier examples are necessary. Outliers are provided either in a validation stage, or they are generated according to an assumed distribution. In principle, an one-class classifier is trained with a fixed target rejection rate ε_t^{tr} (or the threshold θ) for which the threshold is determined. This classifier is then optimised for one point on the ROC curve.

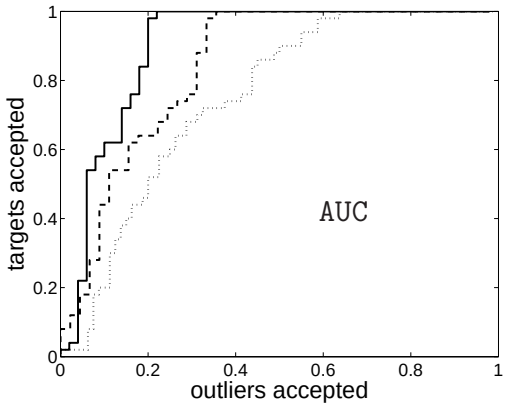


Figure 1.5: Receiver Operator Characteristics (ROC) curves. The area corresponds to the AUC measure associated with the ROC curves.

In order to compare the performance of various classifiers, the AUC measure can be used [Bradley, 1997]. It computes the Area Under the Curve (AUC), which is the total performance of a one-class classifier integrated over all thresholds:

$$\mathsf{AUC} = 1 - \int_0^1 \varepsilon_o(\varepsilon_t)\, d\varepsilon_t = 1 - \int_0^1 \int_{\mathbb{R}^N} \mathcal{I}(d(\mathbf{x}|X_t) < \theta)\, d\mathbf{x}\, d\theta, \tag{1.5}$$

where θ is estimated on the target set. An AUC value smaller than 0.5 indicates that a particular one-class classifier is worse than a random guess. The larger AUC, the better the one-class classifier is. For instance in figure 1.5, the solid curve indicates better performance since the corresponding AUC value is larger than the AUC value for the dashed and doted curve. In

practice, ε_t^{tr} may be limited to a tighter range, so the integration above can be performed for a specified region of interest, such as $[0, 0.5]$, for instance. Moreover, as the costs of making wrong decisions may differ, additionally a weighting function may be introduced.

1.3 Outline of the thesis

The thesis is divided into three major parts: **One-class classifiers**, **Model selection in one-class classification** and **Accommodation of unlabelled data to enhance classification performance**.

Part I: **One-class classifiers**
In the first part of the thesis one-class classifier models are discussed. We start by giving an overview of existing one-class classifier models. The models are divided into two groups: probabilistic models and domain-based (or geometry-based) models. Next, we propose two additional domain-based models. The proposed models are based on the minimum volume enclosing ellipsoid algorithm and the minimum spanning tree algorithm. The performance of new models are compared with existing classifiers and conclusions are drawn for what type of problems the proposed models can outperform existing classifiers.

Part II: **Model selection in one-class classification**
In the second part of the thesis we investigate the problem of model selection in the problem of one-class classification. We propose a model selection method based on the ratio of two volumes: the volume of the largest empty N-sphere that can be found inside a one-class classifier divided by the volume of the one-class classifier. To compute the ratio of two volumes several subproblems have been solved. In section 6.1.1 we present a formula to compute a tight approximation of the volume of one-class classifiers consisting of several intersecting N-spheres such as k-means, k-centres and self-organising maps. The proposed approach can tightly approximate the volume of a given classifier in any number of dimensions. In the same section, we derive a formula to compute the volume of a spherical cap in arbitrary number of dimensions and present a method to check whether more than two N-spheres have a common region. Next, in section 6.1.2 we propose an algorithm to find the largest empty N-sphere in one-class classifiers consisting of N-spheres. Here, we propose a method to check whether a N-sphere is entirely inside a set of intersecting N-spheres. Section 6.1.3 presents an explanation why the presented algorithm does not work for spherical kernel based one-class classifiers such as `SVDD` and *oc*-`SVM`. Finally, an approximate largest N-sphere search algorithm is presented in section 6.1.4 that is applicable to any one-class classifiers. The proposed algorithms are tested on `UCI` repository datasets and the presented model selection method is compared with existing methods.

Part III: **Accommodation of unlabelled data to enhance classification performance**
In the third part of the thesis, we investigate techniques to accommodate unlabelled data into training of classifiers. First, we formulate and investigate active learning techniques. In active learning one is interested in sampling of unlabelled data in such a way that after revealing the true labels of unlabelled objects improves the performance of the classifier significantly. We proposed two active sampling function: `vila` which is based on variation in labelled assignments and `pdc` which is based on positive density corrections.

Next, we investigate semi-supervised learning techniques. In semi-supervised learning one

improves the classification performance by adding information about the distribution of unlabelled objects to the training of classifiers. We propose a semi-supervised algorithm based on the stability of soft-labels. The density-based classifiers such as `LDA`, `QDA`, mixture of Gaussians and `Parzen` density estimator is redefined to incorporate soft-labels of unlabelled data.

The thesis is ended by some final conclusions and an outlook to possible feature research directions.

1.4 Main contributions

In this work, we focus on recognition and learning problems in pattern recognition. The first and second part of the thesis focus on recognition problems, called the one-class classification and the third part focuses on learning problems, like active and semi-supervised learning. The main contributions of the thesis is a set of new algorithms for identification and recognition purposes.

Recognition problems

In chapter 3, we propose a new one-class classifier based on the concept of the Minimum Volume Enclosing Ellipsoid (`MVEE`). A target class is enclosed in the `MVEE`. Three optimisation algorithms are analysed by us: a description of a target class by the `MVEE`, a more robust version of the `MVEE` where outliers are expected in a training set, finally, an estimation of the `MVEE` where labelled outlier objects are available during training. In chapter 4, we propose a new one-class classifier based on a structure of a graph. In particular, we introduce a classifier based on the Minimum Spanning Tree algorithm called a Minimum Spanning Tree Data Description (`MST_DD`).

In part II of the thesis we propose a model selection criterion based on a ratio of two volumes: the volume of the largest empty N-sphere that can be found inside a one-class classifier and the volume of the one-class classifier. In the following sections we introduce several algorithms to compute this ratio:

In section 6.1.1, we present a formula to compute a tight approximation of the volume of one-class classifiers consisting of several intersecting N-spheres. In 6.1.2, we propose an algorithm to find the largest empty N-sphere in one-class classifiers consisting of N-spheres and a method to check whether a N-sphere is entirely inside a set of intersecting N-spheres. In appendix B.1, we derive a formula to compute the volume of a spherical cap created by two overlapping N-spheres. In section 6.1.4, an approximate largest N-sphere search algorithm inside an arbitrary one-class classifiers is introduced.

Learning problems

In part III of the thesis, we study active and semi-supervised learning. In section 7.1, we introduce a new active learning method based on positive density corrections `pdc`. In section 7.2, we introduce a new active learning method based on variation in labelled assignments (`vila`). Chapter 8, introduces three new query diversification algorithms, in active-learning, based on distances, densities and angles between objects. In chapter 9, semi-supervised algorithm for a Parzen and other density based classifiers is introduced. The algorithm estimates soft labels and a smoothing parameter using labelled and unlabelled objects.

Credits. This thesis contains work that has been published or submitted before. Robert Duin, David Tax, Dick de Ridder and Serguey Verzakov are acknowledge for the discussions on all types of pattern recognition and machine learning issues, which resulted in common publications. Most of the experiments have been conducted using `PRTools`, `DD_tools` and own routines.

Part I: One-class classifiers

You can ask one to classify objects,
But you can not order to recognise them.

Summary of Part I: One-class classifiers

In this part we take a closer look at classification models that can be used to describe a target class in the one-class classification problem. We start from a general overview of existing one-class classification models in chapter 2. The chapter is divided into two sections. The first section describes statistically-based one-class classifiers and the second, domain-based one-class classifiers. In following chapters, we introduce new domain-based one-class classifiers. In chapter 3, a one-class classifier based on the Minimum Volume Enclosing Ellipsoid (`MVEE`) algorithm is introduced. In chapter, 4 we introduce a one-class classifier based on the minimum spanning tree algorithm, Minimum Spanning Tree Data Description (`MST_DD`).

Chapter 2

Introduction to one-class classification models

The problem of novelty detection has recently gained a lot of attention as it can be identified in many practical applications, [Tax et al., 2006, Markou and Singh, 2003a, Markou and Singh, 2003b]. This problem can be approached in the framework of one-class classification [Moya et al., 1993, Tax and Duin, 1999, Manevitz and Yousef, 2001, Pękalska et al., 2003, Koppel and Schler, 2004], in which the specified target class has to be distinguished from all other possible examples, which are accounted for the outlier class. It is usually assumed that only target examples are available during training. The reason for the absence of outlier examples can be the very high measurement costs or the low frequency of an event, as for instance in the case of a nuclear power plant failure or a rare medical disease. Another reason lies in either too weak or too broad definition of the outlier class. For instance, if the target class consists of healthy people, the outlier class refers to the class of people carrying all types of diseases. Even when available, outlier examples can not always be trusted, as they are badly represented, with unknown priors and ill-defined distributions. Therefore, the area of interest in one-class classification covers all the problems of novelty detection by the recognition of a specified and reasonably well sampled and described target class from all kinds of anomalies, as weakly defined in an outlier class. The applications are any type of fault detection [Ypma and Duin, 1998], abnormal behaviour, rare illnesses [Tarassenko et al., 1995], authorship verification [Koppel and Schler, 2004], etc.

The principles behind many two- or multi-class classifiers can be used for solving one-class classification problems. The most common approach is probabilistic [Bishop, 1995]. Basically, the target class is modelled by some probability density function. Specifying a suitable threshold allows one to determine the class boundary. A test sample is judged to be a member of the target class if the estimated probability is higher than the given threshold. This can be realised by a parametric method such as the mixture of Gaussians [Sain et al., 1999] or even a single Gaussian equipped with a threshold [Chow, 1970], or by a non-parametric method like the Parzen density estimator [Parzen, 1962] or k-nearest neighbour estimators [Knorr et al., 2000]. Other popular approaches are neural networks, including auto-encoders [Japkowicz, 1999] or self-organising maps [Parra et al., 1996] and various clustering techniques such as k-means [Jiang et al., 2001] or k-centres [Hochbaum and Shmoys, 1985].

Alternative solutions to the one-class classification problem have been proposed, that do not use a probabilistic approach. These methods are based on minimisation of the volume of a target class domain. This is realised by the use of linear programming [Campbell and Bennett, 2000, Lanckriet et al., 2003, Pękalska et al., 2003] and quadratic pro-

gramming [Tax and Duin, 1999, Schölkopf et al., 2000a]. In particular, the support vector one-class classifier, the Support Vector Data Description (SVDD) has been introduced in [Tax and Duin, 1999]. In the input space, this classifier finds the smallest N-sphere that encloses all objects from the target class. Other flexible descriptions are enabled by the use of kernels in the spirit of the support vector machines [Vapnik, 1998]. A similar method, the one-class SVM (*oc*-SVM), has been proposed in [Schölkopf et al., 2000a], which uses a hyperplane to maximally separate data from the origin. In contradiction to the statistical approaches, domain-based classifiers are not driven by the frequency of appearances of objects, in a representation space, but by geometrical shape of a domain of a target class.

Next, we describe both: statistical and domain-based approaches to one-class classification problems.

There are two basic approaches in characterising the target data. The first one uses a statistical approach, often involving a density estimation of the target class. This assumes that the target data is sampled well, and that low density areas in the training set indicate that these areas have a low probability of containing target objects. However, in cases that a true sampling is hard to obtain, another approach is required.

The second approach uses a domain-based approach for the characterisation of the target data. This method tries to describe a boundary around the target class, such that the captured volume is minimised. When the uniform outlier distribution is assumed this means that the chance of accepting an outlier object is minimised. The advantage of this approach is that no probability density on the targets have to be estimated.

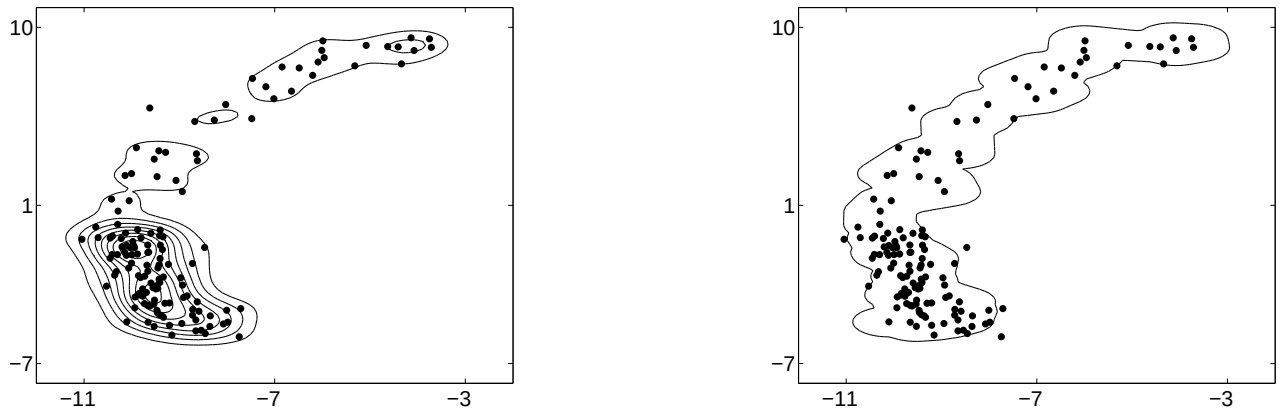


Figure 2.3: The two one-class classifier models: (Left) the statistical approach and (Right) the domain-based approach. The threshold for both classifiers was set to $\theta = 0.05$.

In figure 2.3 examples of two approaches are shown on $2D$ dataset with a high data density in the bottom part. In the left subplot a density estimator is shown. The high density cluster causes a focus on this region. Due to the low density of the tail on the other end of the distribution, this data is likely to be considered outliers. The right subplot shows a domain-based method. This approach ignores the local variation in density. The decision boundary by this method follows the shaped distribution but does not focus more specifically on the high density regions. Depending on the high density cluster being a sampling artifact, or a genuine target density feature, the second respectively the first approach is to be preferred.

2.1 Statistics-based one-class classifiers

The most common one-class classifiers are density estimators. The density methods estimate the complete probability density of the target class, given by a training set, and threshold this density [Chow, 1970, Ben-David and Lindenbaum, 1997]. This approach is often taken to detect outlier objects in a supervised classification. Here the classifier depends on the class conditional probabilities and thus the density of the target class is available. The drawback of the density models is that estimating densities is a hard problem, especially when a limited amount of data is available. Either the density method imposes a restrictive model on the data, parametric modelling, which results in a large bias when the model does not fit the data, or the model is very flexible and requires a large sample size to reliably fit all the free parameters, i.e. non-parametric modelling. On the other hand, when a sufficient number of objects is available, a good performance can be achieved.

In practice one often approximates the target density by much simpler models. In these models a full density estimate is avoided. Instead the target data is characterised by cluster centres or subspace models. We first discuss some density estimators and then the clustering methods and subspace methods are treated.

2.1.1 Density-based classifiers

Gaussian density estimation

The most simple statistical model is the normal density [Bishop, 1995]. According to the Central Limit Theorem, this model is correct when we assume that objects from one class originate from one prototype and are disturbed by a large number of small independent variations. For this density model the target-class conditional probability $p_G(\mathbf{x}|\omega^{(t)})$ that a new object $\mathbf{x}$ belongs to the target class is express as:

$$p_G(\mathbf{x}|\omega^{(t)}) = \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{2.1}$$

the mean $\boldsymbol{\mu}$ and covariance matrix Σ have to be estimated, therefore the Gaussian density estimator has $(N + \frac{N(N+1)}{2})$ free parameters (note that the covariance matrix is symmetric). Although the method is very simple, it imposes a strict unimodal and ellipsoidal density model on the data. For high dimensional data this model also suffers from very large covariance matrices, that are hard to estimate since the computation of their inverse becomes easily ill-defined. An example of a classifier based on a Gaussian density is shown in figure 2.4(a).

The standard estimators for the mean and covariance matrix are not robust against outliers. Several alternative methods have been proposed in the literature [Kosinski, 1999]. One alternative is the Minimum Covariance Determinant (`MCD`) method by [Rousseeuw and van Driessen, 1999]. Here, only a user-specified fraction of the training objects is used for fitting a Gaussian distribution. Only that subset is used that results in the smallest determinant of the covariance matrix. Because the rest of the data is not used, this procedure is very robust and even a high fraction of outliers does not deteriorate the solution.

Mixture of Gaussians

An extension of the Gaussian distribution is the mixture of Gaussians; see figure 2.4(b) for an example, which is a linear combination of Gaussian distributions:

$$p_{MoG}(\mathbf{x}|\omega^{(t)}) = \frac{1}{\sqrt{(2\pi)^N}} \sum_{j=1}^{\gamma} \alpha_j \frac{1}{\sqrt{\det(\Sigma_j)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^T \Sigma_j^{-1} (\mathbf{x}-\boldsymbol{\mu}_j)\right) \tag{2.2}$$

where α_j are the mixing coefficients. Generally, it has a smaller bias than the single Gaussian distribution, but requires more data. The number of clusters (γ) has to be defined beforehand and determines the complexity of the model. The means and covariances have to be estimated [Sain et al., 1999, Lauer, 2001]. The number of free parameters to be estimated is ($\gamma(N + \frac{N(N+1)}{2})$). To reduce the number of free parameters, often just diagonal covariance matrices are assumed, ($2\gamma N$). Expectation Minimisation (EM) can be used to estimate the parameters [Bishop, 1994, Bishop, 1995], where the user also has to supply the maximal number of iterations in the optimisation.

Parzen density estimator

A third method is the Parzen density estimation [Parzen, 1962, Yeung and Chow, 2002, Nunez-Garcia et al., 2003]. The estimated density is a mixture of kernels centred on the individual training objects:

$$p_P(\mathbf{x}|\omega^{(t)}) = \frac{1}{n} \sum_{i=1}^{n} K(\mathbf{x}; \mathbf{x}_i, \gamma) \tag{2.3}$$

The most often used kernel is a Gaussian kernel with diagonal covariance matrices; see figure 2.4(c). Training the Parzen density consists of the determination of the width of the kernel γ. γ can be optimised by maximising the likelihood [Duin, 1976]. The fixed width in each feature direction means that the Parzen density estimator is sensitive to the scaling of the data, especially for lower sample sizes. Because this method contains just a single parameter, the optimisation can be applied even with a relatively small training set.

The Parzen density estimator can be simplified by assuming independence of the features, similar to the Naive Bayes classifiers in supervised classification [Hastie et al., 2001]. For each feature x_j a one-dimensional probability density is estimated. The total probability density is a product of the individual feature densities; see figure 2.4(d). Obviously, this method ignores the feature correlations and therefore may fail to describe the target class well. On the other hand, it is very likely that sufficient data is available to estimate one-dimensional distributions relatively reliably. For high dimensional data, and data without very strong feature correlations, this density estimator works well [Tax et al., 2006].

More advanced density estimators have been used for outlier detection, like hierarchical probabilistic models [Baker et al., 1999] or Hidden Markov Models [Yeung and Ding, 2003]. They aim to exploit some characteristics in the dataset, or to use specific model assumptions, to make the density estimation feasible and efficient. As they are application dependent, we do not discuss them further.

2.1.2 Clustering-based classifiers

When density estimation is not feasible, one can approximate the target class by a simpler model. Instead of a density this model captures the data structure and new objects are projected onto this model. The reconstruction error, the difference between the original object $\mathbf{x}$ and the projected object $\mathbf{p}(\mathbf{x})$, indicates the resemblance of a new object to the original target distribution. We denote this distance by $d(\mathbf{x}, X_t)$, where X_t is the training set. Two types of models are distinguished: the clustering approaches and the subspace approaches. The difference between $\mathbf{x}$ and $\mathbf{p}(\mathbf{x})$, called the reconstruction error, is used to characterise how well an object fits the model, and how likely this object is an outlier.

The clustering-based classifiers include data clustering or compression methods like Learning Vector Quantisation (`LVQ`), k-means clustering [Bishop, 1995, Jiang et al., 2001] or a self-organising map (`SOM`) [Kohonen, 1995, Marsland, 2001]. In these methods, the target class is characterised by a few prototype objects $\mathbf{c}_k \in X_t$. The minimum distance from the test object to the nearest prototype object is often used as a distance measure between a new object and the target class:

$$d(\mathbf{x}, X_t) = \min_{\mathbf{c}_k \in X_t} \|\mathbf{x} - \mathbf{c}_k\| \tag{2.4}$$

The methods use different approaches to obtain the prototype locations. `LVQ` and k-means place the prototypes as best as possible in the mean-square-error sense by iteratively updating the prototype positions; figure 2.4(e). k-means clustering uses an expectation-maximisation algorithm to update the prototypes [Dempster et al., 1977, Bishop, 1995], while `LVQ` applies a gradient descent type method [Haykin, 1999].

`SOM` incorporates an extra constraint to form a low-dimensional manifold, often 2 or 3 dimensional, which makes it possible to visualise high dimensional data in a 2 or $3D$ plot. Because distances to the prototypes are used, these methods are sensitive to rescaling of features. The number of prototypes should be given by the user and sometimes also the number of training epochs is required.

Nearest neighbour

A method that avoids the optimisation of the prototype locations, is the nearest neighbour method. It uses all objects in the training set as prototypes [Knorr et al., 2000, Harmeling et al., 2005], but obviously, some condensing can be applied [Hart, 1968]. When many more objects are available, one can utilise not only the nearest prototype, but also take information of the next nearest neighbours into account. We denote the k nearest neighbour of an object $\mathbf{x}$ in X_t as $\mathbf{x}_{(k)}$. One can define several distance measures:

$$d_k(\mathbf{x}, X_t) = \|\mathbf{x} - \mathbf{x}_{(k)}\| \qquad \text{distance to the } k\text{-th nearest neighbour} \tag{2.5a}$$

$$d_a(\mathbf{x}, X_t) = \frac{1}{k}\sum_{i=1}^{k} \|\mathbf{x} - \mathbf{x}_{(i)}\| \quad \text{average distance to the } k \text{ nearest neighbours} \tag{2.5b}$$

$$d_\mu(\mathbf{x}, X_t) = \|\mathbf{x} - \frac{1}{k}\sum_{i=1}^{k} \mathbf{x}_{(i)}\| \quad \text{distance to the average of the } k \text{ nearest neighbours} \tag{2.5c}$$

The first measure (2.5a) only considers the distance to the k-th nearest neighbour. Measures (2.5b) and (2.5c) include some information of closer neighbours. (2.5b) averages the distances

of the neighbours, making it more sensitive to density changes in the data; see figure 2.4(f). (2.5c) computes the distance to the average of the neighbours. It therefore becomes sensitive to the direction in which the neighbours are distributed. Obviously, when $k = 1$, all three methods are identical. It appears that these methods work well in higher dimensional feature spaces [Harmeling et al., 2005]. The density estimation in these spaces fails, but the nearest neighbour distances are still indicative for outliers.

To incorporate information of the local density, one can look at the local nearest neighbour distances. One can compare the distance of a new object $\mathbf{x}$ to its nearest neighbour $\mathbf{x}_{(1)}$ in X_t, with the nearest neighbour distance of this object [Tax, 2001]. When this first distance is larger than the second, object $\mathbf{x}$ is likely an outlier. Indicating the first nearest neighbour by $\mathbf{x}_{(1)}$ and its nearest neighbour by $\mathbf{x}_{(1)}(\mathbf{x}_{(1)})$, this classifier can be described by:

$$d_{NN}(\mathbf{x}, X_t) = \frac{\|\mathbf{x} - \mathbf{x}_{(1)}\|}{\|\mathbf{x}_{(1)} - \mathbf{x}_{(1)}(\mathbf{x}_{(1)})\|} \tag{2.6}$$

This method is very noise sensitive, and a single outlier in the training set results in a very high false positive rate. On the other hand, it can work with a very low sample size and does not contain free parameters to optimise.

2.1.3 Subspace-based classifiers

The second type of reconstruction methods includes the subspace models. When the data is very high dimensional this method might be preferred since the nearest neighbour distance loses its meaning. [Beyer et al., 1999]. In such cases it can often be assumed that the target data is distributed in subspaces of much lower dimensionality.

For data in a linear subspace often Principal Component Analysis (`PCA`) [Jolliffe, 1986] is used. `PCA` finds the orthogonal subspace which captures the variance in the data as best as possible, in the square error sense. When the basis vectors $\mathbf{w}$ are stored in an $N \times N'$ matrix, object $\mathbf{x}$ is reconstructed onto this subspace by:

$$\mathbf{p}(\mathbf{x}) = \mathbf{w}(\mathbf{w}^T\mathbf{w})^{-1}\mathbf{w}^T\mathbf{x} = P\mathbf{x}, \tag{2.7a}$$

$$d_{PCA}(\mathbf{x}, X_t) = \|\mathbf{x} - \mathbf{p}(\mathbf{x})\| \tag{2.7b}$$

The number of basis vectors N' is optimised to preserve a certain, user defined, fraction of the variance in the data. Similar to (2.4), the Euclidean distance from the original $\mathbf{x}$ and the reconstructed object $\mathbf{p}(\mathbf{x})$ is used as resemblance measure [Tax, 2001, Shyu et al., 2003]. An example of the `PCA`-based classifier is shown in figure 2.4(g).

There are several non-linear subspace methods. Next to the self-organising map, there are the auto-encoders, auto-associative or Diabolo networks [Baldi and Hornik, 1989, Surace et al., 1997]. These are neural network approaches that learn a low dimensional non-linear representation of the data. A standard feedforward neural network is trained to reproduce the input patterns $\mathbf{x}$ at its output layer. One of their hidden layers contains a small number of hidden units which works like an information bottleneck. Similar to (2.4), the difference between the input $\mathbf{x}$ and output $\mathbf{p}(\mathbf{x})$ defines the reconstruction error. To obtain a small reconstruction error, the networks are forced to train a very compact representation of the data. When just one hidden layer is used, a linear Principal Component solution is found [Bourlard and Kamp, 1988]. Using more hidden layers with non-linear transfer functions the neural network describes a more flexible, non-linear subspace. Although this method is very

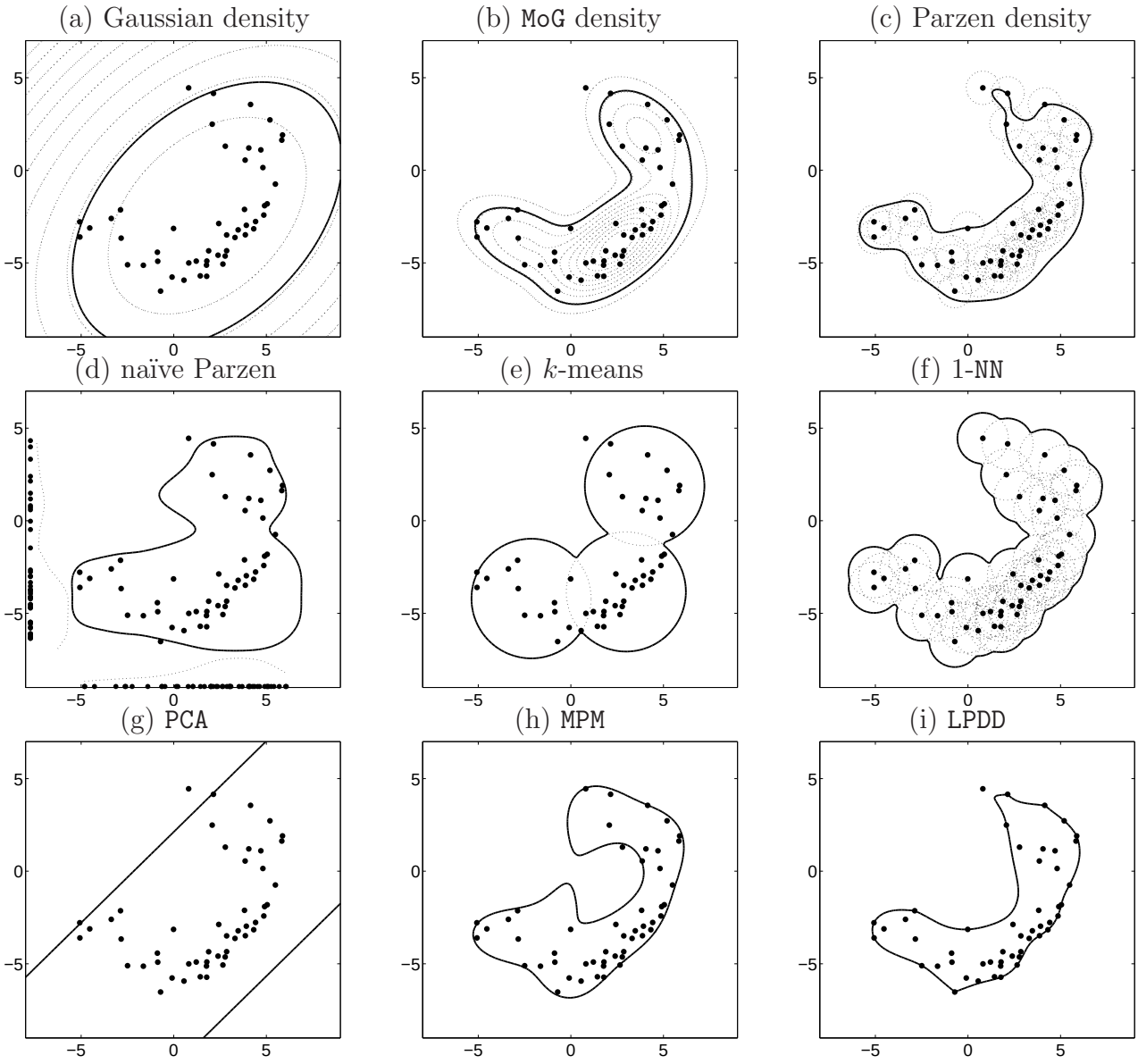


Figure 2.4: Examples of statistics and domain driven one-class classifiers (continuous line). The principles behind some one-class classifiers are denoted by dotted lines. The threshold was set to reject $\theta = 0.01$ of the target class. For 1-NN the threshold was set on the averaged nearest neighbour distance.

flexible, it requires a predefined number of layers, the sizes of the layers and learning rates. It can happen that the resulting decision boundary is actually not closed around the target class.

2.2 Domain-based one-class classifiers

In the domain-based methods just the boundary of the target class is determined. Because hardly any information on the statistics of the target data is used, these methods become insensitive to the specific sampling and density of the target class. Such methods describe the target class boundary, or the domain, and not the class density. This can be a large advantage for applications where only the domain in feature space can be indicated, but the

genuine density distribution is unknown. By their focus on the decision boundary, these methods are often most influenced by the outliers in the training set, and are very dependent on the suitable scaling of the features.

k-centres one-class classifier

The first approach is the k-centres method that covers the dataset with balls with equal radius [Hochbaum and Shmoys, 1985, Ypma and Duin, 1998], it has a resemblance with the covering numbers by Kolmogorov [Kolmogorov and Tikhomirov, 1961], also called the k-medoids clustering [Hastie et al., 2001]. The k ball centres $\mathbf{c}_k$ are placed such that the maximum distance of training objects to the nearest centre is minimised. The distance from the target class to a new object is measured as a distance to the closest centre:

$$d_{k-c}(\mathbf{x}, X_t) = \min_{\mathbf{c}_k \in X_t} \|\mathbf{x} - \mathbf{c}_k\| \tag{2.8}$$

To optimise $\mathbf{c}_k$ a forward search after a random initialisation is used and like in the mixture of Gaussians the number of balls, k, has to be set beforehand. Unfortunately, by the equal radii for all of the balls and the difficult optimisation, it appears that this procedure is very unstable. Usually the best solution over a set of random initialisations is chosen. This requires that the user has to supply a maximum number of trials.

`SVDD`

A more flexible approach is the Support Vector Data Description (`SVDD`) method. It finds the hypersphere, parameterised by a centre $\mathbf{a}$ and a radius R, around the dataset that has minimal volume [Tax and Duin, 2004]. Its optimisation is basically:

$$\min_{\mathbf{a},R} \quad R^2 \tag{2.9a}$$

$$\text{s.t.} \quad \|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2, \quad i = 1, \ldots, n \tag{2.9b}$$

It is the one-class variant of the support vector machine [Vapnik, 1998] and it has a similar type of quadratic optimisation problem to solve during its training. Analogous to the normal support vector machine, a dual formulation can be derived that is completely in terms of inner products. By replacing the normal inner products $(\mathbf{x}_i \cdot \mathbf{x}_j)$ by a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ the flexibility of the model can be increased. The distance of a test object $\mathbf{x}$ to the sphere centre, in feature space, can then be computed by:

$$d_{SVDD}(\mathbf{x}, X_t) = K(\mathbf{x}, \mathbf{x}) - 2\sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + \sum_{i,j} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.10}$$

The weights α_i are obtained by quadratic optimisation [Tax, 2001]. To make the `SVDD` more robust, slack variables can be introduced that allow to have a few training objects outside the boundary. The `SVDD` can be extended to include outlier objects in its training. Instead of using only the constraints that the target objects should be inside the hypersphere, constraints can be added to force the outliers outside the hypersphere. In that case we obtain a classifier, with the extra feature that it encloses the target class completely [Tax, 2001]. We describe `SVDD` in more details in the next chapter.

One-class Support Vector Machine

When domain-based classifiers are applied to data represented in a vector space, they should envelop the target class from all sides because outliers can be expected in all directions. This constrains the geometrical shape of the classifier. In most cases these classifiers are ball-shaped, or consist of several balls. This shape constraint can be removed, when one transforms the data to a new representation. Consider for instance a similarity representation. Objects are now not represented by externally defined features, but by their relative similarity to other objects. The radial basis function kernel in the support vector machine is an example. Objects close to the target data have high similarities, and are represented far away from the origin of the similarity space. Outliers on the other hand, are mapped close to the origin of this similarity space. Outliers that are widely scattered in the original feature space, are therefore focused in a very specific corner in the similarity space by this transformation. This simplifies the construction of a classifier significantly.

This approach to outlier detection is followed by [Schölkopf et al., 2001, Manevitz and Yousef, 2001, Rätsch et al., 2002]. One tries to find the optimal linear hyperplane that separates the target data with the largest margin from the origin. It results in a optimisation problem that is very similar to the support vector machine or the `SVDD`, and is also called the One-Class Support Vector Machine, *oc*-`SVM`. It can be kernelised, and in practice most often the RBF kernel is applied. But other similarity measures, like a Hausdorff distance kernel to compare images [Barla et al., 2002], can be used as well. It appears that when one uses an RBF kernel in the `SVDD`, it reduces to the *oc*-`SVM`. In that case $K(\mathbf{x}, \mathbf{x}) = 1$ and the classifier becomes linear in terms of $K(\mathbf{x}, \mathbf{x}_i)$:

$$d_{oc-SVM}(\mathbf{x}, X_t) = -\sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) \tag{2.11}$$

Furthermore, the quadratic optimisation of the classifier can be simplified to a linear programming approach, resulting in the LP one-class classifier [Campbell and Bennett, 2000].

Single-Class Minimax Probability Machine

A similar approach is taken in the Single-Class Minimax Probability Machine (`MPM`) [Lanckriet et al., 2003]; see figure 2.4(h). Here a hyperplane is fitted such that the Mahalanobis distance from the origin to the closest point on the hyperplane is maximised. The Mahalanobis distance uses the covariance matrix of the target data, thus taking some information on the distribution of the target class into account. The final classifier is identical to (2.11), but the weights α_i are not sparse. The optimisation on the other hand appears to be much simpler, it requires the inverse of an adapted kernel matrix.

Linear programming data description

When instead of similarities, like the RBF kernel representation, distances between objects $\mathbf{x}$ and the training set X_t are given, the outliers are not close to the origin, but far away. To obtain a tight description of the target class in this case, one should put a decision boundary as close as possible to the origin. One can thus minimise the L_1 distance ρ of a hyperplane to the origin in the distance representation, and the Linear Programming Data Description (`LPDD`)

is obtained [Pękalska et al., 2003]; see figure 2.4(i). It results in the following formulation:

$$\min \quad \rho + \frac{1}{\nu\, n} \sum_{i=1}^{n} \xi_i \tag{2.12a}$$

$$\text{s.t.} \quad \mathbf{w}^T \|\mathbf{x}_i - \mathbf{x}_j\|_1 \leq \rho + \xi_i, \quad i, j = 1, 2, \ldots, n \tag{2.12b}$$

$$\sum_j w_j = 1,\ w_j \geq 0,\ \rho \geq 0,\ \xi_i \geq 0. \tag{2.12c}$$

Here, ν is a user defined complexity parameter, indicating how strictly all training objects should be classified as target objects. The final classifier is linear in the distance representation:

$$d_{LPDD}(\mathbf{x}, X_t) = \sum_i w_i \|\mathbf{x} - \mathbf{x}_i\|_1 - \rho \tag{2.13}$$

In order to make the classifier robust against outliers in the training set i.e. objects with large values for $\|\mathbf{x} - \mathbf{x}_i\|_1$, the distances can be transformed by the sigmoid function $\text{sigm}(\|\mathbf{x} - \mathbf{x}_i\|_1)$, such that for large values, the resulted value is bounded. For the sigmoid a suitable parameter for its slope has to be defined.

We have not described all proposals for a one-class classifier in the literature, but most of the principles and assumptions behind their architectures and assumptions have been covered. The next sections introduce some new one-class classifiers. We also compare performances of one-class classifiers for various types of data.

Chapter 3

Minimum volume enclosing ellipsoid data description

In this section we propose to use the minimum volume enclosing N-ellipsoid (MVEE)[1] as a description of a target class. We investigate several variants of such descriptor. We start from determining the MVEE based only on given training-target objects. The MVEE is estimated as the smallest volume N-ellipsoid that encloses all target objects. Secondly, the robust estimation of the MVEE is investigated. Therefore, we assume that some outliers are present in the training data. By introducing slack variables we allow some training objects to be outside the ellipsoid . Finally, we investigate the estimation of the MVEE when few labelled outlier objects are available. We introduce slack variables for both classes and we minimise the volume of MVEE is such a way that the sum of slacks is also minimised. If objects from target or outlier class are misclassified the value of their slack variables are proportional to their distances to the surface of the N-ellipsoid. We allow misclassifications of objects from both classes if the volume of the N-ellipsoid is sufficiently small. An additional parameter is introduced that determines the trade-off between the volume of an N-ellipsoid and the sum of slack variables.

We expect that objects from the target class are close in the representation space $\mathbb{R}^N$. This standard assumption is called the compactness hypothesis [Duin, 1999] and it characterises well behaved representations. The second assumption we make is a more strict one: we assume unimodality of the target class. Therefore, this suggests that we can enclose objects from the target class in some kind of a hull, possibly an N-sphere. However we also want our model to be scale invariant. This is why we use the affine deformations of an N-sphere, which is an N-ellipsoid.

The minimum-volume enclosing N-ellipsoid problem has been studied for over 50 years. As early as 1948 (possibly even earlier), [John, 1948] discussed this problem in his work on optimality conditions. [Barnes, 1982] provides an algorithm for this problem based on a matrix eigenvalue decomposition. [Khachiyan and Todd, 1993] first used interior-point methods in developing an algorithm and a complexity bound for the closely related maximum-volume inscribed N-ellipsoid problem (MVIE), together with a linear reduction from MVEE to MVIE; the complexity of their algorithm for finding an ϵ-optimal N-ellipsoid is $\mathcal{O}(n^{3.5}\ln(\frac{nR}{\epsilon})\ln(\frac{N\ln R}{\epsilon}))$ arithmetic operations, where N is the dimensionality of data and n the number of objects. Here, R is defined such that the convex hull of the given points contains the unit N-sphere centred at the origin and is contained in the concentric N-sphere of a given radius R, and ϵ is a relative measure of non-optimality. [Nesterov and Nemirovskii, 1994] obtain a complex-

[1]Similar to a N-sphere an N-ellipsoid denotes an ellipsoid in N dimensions.

ity bound of $\mathcal{O}(n^{3.5}\ln(\frac{nR}{\epsilon}))$ operations, and more recently [Khachiyan, 1996] has reduced this to $\mathcal{O}(n^{3.5}\ln(\frac{n}{\epsilon}))$ operations. [Zhang, 1998] presents interior-point algorithms for the maximum volume inscribe N-ellipsoid, based on various equation system reduction schemes. In 2003, [Zhang and Gao, 2003] extended their earlier results and compare different practical algorithms for the maximum-volume inscribed N-ellipsoid problem. [Vandenberghe et al., 1998] and [Toh, 1999] also consider the minimum volume N-ellipsoid problem as a special case of the more general maximum determinant problem.

In this section we investigate several variations of the maximum determinant formulation of the `MVEE` problem [Toh, 1999], to be applied to the one-class classification problem. In particular, we introduce the robust estimation of the `MVEE` and the estimation of an ellipsoid when some labelled outlier objects are available.

3.1 Minimum Volume Enclosing Ellipsoid

Our concern is with covering n given points $X_t := \{\mathbf{x}_i, \mathbf{x}_i \in \mathbb{R}^N, i =, 1 \ldots, n\}$ with an ellipsoid of the minimum volume. To avoid trivialities, we make the following assumption, which guarantees that any ellipsoid containing $\{\mathbf{x}_1, \mathbf{x}_2 \ldots \mathbf{x}_n\}$ can be computed in $\mathbb{R}^N$:

Assumption 3.1 *There is a subset of objects $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\} \subset X_t$ which is affinely independent.*

This assumption is necessary since the computation of not fully dimensional ellipsoid is not trivial [Boyd and Vandenberghe, 2003].

We now state the formal definition of an ellipsoid.

Definition 3.1 *An ellipsoid $\mathcal{E} \subseteq \mathbb{R}^N$ is a set described by a centre $\mathbf{c} \in \mathbb{R}^N$ and an $N \times N$ symmetric positive definite matrix E such that*

$$\mathcal{E}_{E,\mathbf{c}} := \{\mathbf{x} \in \mathbb{R}^N | (\mathbf{x} - \mathbf{c})^T E(\mathbf{x} - \mathbf{c}) \leq 1\} \tag{3.1}$$

where E determines shape and orientation of the N-ellipsoid $\mathcal{E}_{E,\mathbf{c}}$. In particular, the axes of $\mathcal{E}$ are eigenvectors of E and the length of the axes is given by $[\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_N}]$, where $[\lambda_1, \ldots, \lambda_N]$ are the corresponding eigenvalues of the matrix E.

We denote the positive definiteness of E by $E \succ 0$, this is equivalent to $\mathbf{x}^T E\mathbf{x} > 0,\ \forall \mathbf{x} \in \mathbb{R}^N$ or $\lambda_i > 0, \forall_i$. When E is not positive definite ($E \not\succ 0$) or semipositive definite ($E \not\succeq 0$), equation (3.1) describes any quadratic set. As we note further in this section and appendix A, the positive definite cone $\mathbb{R}^N_+$ of feasible solutions of E helps us to model such a descriptor. We can also note that the ellipse $\mathcal{E}$ induces a norm on $\mathbb{R}^N$ via $\|\mathbf{x}\|_\mathcal{E} := \sqrt{\mathbf{x}^T E\mathbf{x}}$. Therefore, Euclidian distances are treated differently in different directions.

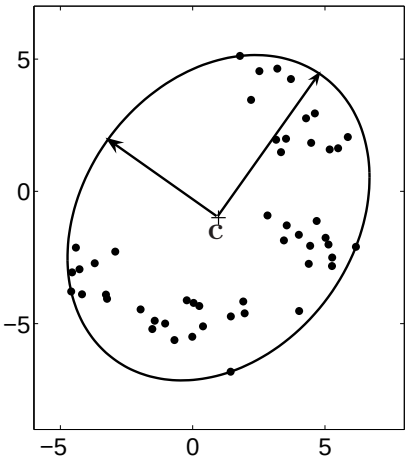


Figure 3.1: Minimum volume enclosing ellipsoid.

The volume of $\mathcal{E}_{E,\mathbf{c}}$ is given by the following formula [Grötschel et al., 1998]:

$$V_{\mathcal{E}_{E,\mathbf{c}}} = \frac{\pi^{\frac{N}{2}}}{\Gamma(\frac{\pi}{2}+1)} \frac{1}{\sqrt{\det(E)}} = \frac{\pi^{\frac{N}{2}}}{\Gamma(\frac{\pi}{2}+1)} \prod_{i=1}^{N} \frac{1}{\sqrt{\lambda_i}} \tag{3.2}$$

where the first ratio is the volume of the unit N-sphere. By taking the logarithm of equation (3.2):

$$\ln V_{\mathcal{E}_{E,\mathbf{c}}} = \ln\left(\frac{\pi^{\frac{N}{2}}}{\Gamma(\frac{\pi}{2}+1)}\right) - \ln\sqrt{\det(E)} \tag{3.3}$$

Thus we can see that minimising the volume of $\mathcal{E}$ is equivalent to maximising the square root of the determinant of the matrix E. Under the assumption 1, a natural formulation of a minimum volume ellipsoid enclosing all data $\mathbf{x}_i \in X_t$ is:

$$\min_E \quad -\sqrt{\det(E)}, \tag{3.4a}$$

$$\text{s.t.} \quad (\mathbf{x}_i - \mathbf{c})^T E(\mathbf{x}_i - \mathbf{c}) \le 1, \quad \forall i = 1, \ldots, n, \tag{3.4b}$$

$$E \succ 0. \tag{3.4c}$$

Such formulation of the problem of finding minimum volume ellipsoid that encloses all objects $\mathbf{x}_i$ is not a convex program [Nesterov and Nemirovskii, 1994]. However by changing variables:

$$M = \sqrt{E} \qquad \mathbf{z} = \mathbf{c}\sqrt{E} \tag{3.5}$$

we get convex and simpler problem. Where we used the property of determinant $(\det(X))^k = \det(X^k)$, the square root of the matrix X is defined as $\sqrt{X} = V^T\sqrt{D_{[d_{ii}]}}V$, where $\sqrt{D_{[d_{ii}]}}$ is an element-wise square root of eigenvalues.

Now we can rewrite the problem (3.4) as:

$$\min_{M} \quad -\ln\det(M) \tag{3.6a}$$

$$\text{s.t.} \qquad (M\mathbf{x}_i - \mathbf{z})^T(M\mathbf{x}_i - \mathbf{z}) \leq 1, \quad \forall i = 1, \ldots, n \tag{3.6b}$$

$$M \succ 0. \tag{3.6c}$$

which is now a convex program. Such formulation of the problem of finding the minimum volume enclosing ellipsoid can be solve using the conic programming [Lobo et al., 1998].

However, the optimisation (3.6) can be simplified further by mapping data from $\mathbb{R}^N$ to $\mathbb{R}^{N+1}$ and then compute the `MVEE` in the augmented space; see figure 3.2. The mapping is done by adding one additional feature, equal to a constant, to each object $\mathbf{x}_i \in X_t$. In our case the constant equals one. Next, we minimise the volume of the ellipsoid in $\mathbb{R}^{N+1}$ centred at the origin. To show that those two optimisations are equivalent, due to linear transformations, we denote the set of parameters as:

$$\mathbb{R}^N \to \mathbb{R}^{N+1},\ \mathbf{x}_i \to \tilde{\mathbf{x}}_i,\ E \to \tilde{M},\ \mathbf{c} \to \mathbf{0} \tag{3.7}$$

Therefore, the volume of the new ellipsoid $\mathcal{E}_{\tilde{M},\mathbf{0}}$ centred at the origin is optimised. To show that the parameters of the ellipsoid $\mathcal{E}_{E,\mathbf{c}}$ can be computed from the parameters of the ellipsoid $\mathcal{E}_{\tilde{M},\mathbf{0}}$ we decompose $\tilde{\mathbf{x}}$ and the shape matrix $\tilde{M}$ as follows:

$$\tilde{\mathbf{x}}_i = \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix}, \quad \tilde{M} = \begin{pmatrix} s & \mathbf{v}^T \\ \mathbf{v} & H \end{pmatrix} \tag{3.8}$$

to decide whether any object $\mathbf{x}_i$ is inside or outside the ellipsoid $\mathcal{E}_{\tilde{M},\mathbf{0}}$, it is first mapped to $\mathbb{R}^{N+1}$ and then multiplied by the shape matrix $\tilde{M}$:

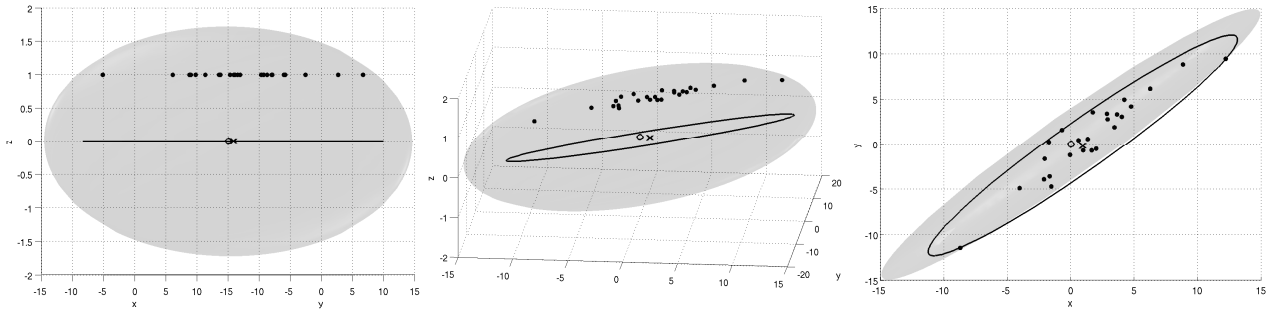
$$\begin{aligned} \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix}^T \begin{pmatrix} s & \mathbf{v}^T \\ \mathbf{v} & H \end{pmatrix} \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix} &= s + 2\mathbf{x}_i^T\mathbf{v} + \mathbf{x}_i^T H\mathbf{x}_i \\ &= s - 2\mathbf{x}_i^T H\tilde{\mathbf{z}} + \mathbf{x}_i^T H\mathbf{x}_i \end{aligned} \tag{3.9}$$

Since the intersection of the ellipsoid $\mathcal{E}_{\tilde{M},\mathbf{0}}$ with the subspace of the data is an ellipsoid in $\mathbb{R}^N$, equation equation (3.9) should satisfy the following inequalities:

$$\begin{aligned} \tilde{\mathbf{x}}_i^T\tilde{M}\tilde{\mathbf{x}}_i &\leq 1 \\ (\mathbf{x}_i - \tilde{\mathbf{z}})^T H(\mathbf{x}_i - \tilde{\mathbf{z}}) + s - \tilde{\mathbf{z}}^T H\tilde{\mathbf{z}} &\leq 1 \\ (\mathbf{x}_i - \tilde{\mathbf{z}})^T \delta^{-1} H(\mathbf{x}_i - \tilde{\mathbf{z}}) &\leq 1, \quad \text{where} \quad \delta = 1 + \tilde{\mathbf{z}}^T H\tilde{\mathbf{z}} - s \end{aligned} \tag{3.10}$$

where we substituted $\mathbf{v}$ by $-H\tilde{\mathbf{z}}$. By comparing inequality (3.10) with inequality (3.6) we can see that:

$$\mathbf{z} = \tilde{\mathbf{z}}, \quad M = \delta^{-1}H$$



(a) (b) (c)

Figure 3.2: Different projections of an ellipsoid computed in $\mathbb{R}^{N+1}$ (here $N = 2$), denoted by gray coloured patch and an ellipsoid in $\mathbb{R}^N$ denoted by black line. The centres of ellipsoids are denoted by $\{\circ\}$ and $\{\mathbf{x}\}$ respectively and the data is denoted by $\{\bullet\}$. (a) View on added feature $\mathbb{R}^N \rightarrow \mathbb{R}^{N+1}$ (b) General view in $\mathbb{R}^{N+1}$. (c) Projection of the ellipsoids and data to original $\mathbb{R}^N$.

Therefore, the minimisation (3.6) can be rewritten as:

$$\min_{\tilde{M}} \quad -\ln\det(\tilde{M}), \tag{3.11a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T \tilde{M} \tilde{\mathbf{x}}_i \leq 1, \qquad \forall \tilde{\mathbf{x}}_i \in \tilde{X}_t, \tag{3.11b}$$

$$\tilde{M} \succ 0. \tag{3.11c}$$

The dual of the minimisation (3.11) is (see Appendix A.2 for a derivation):

$$\max_{\alpha_i} \quad \ln\det \sum_{i=1}^{n} \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T, \tag{3.12a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i = N + 1, \tag{3.12b}$$

$$0 \leq \alpha_i \leq 1, \ \forall i = 1, \ldots n. \tag{3.12c}$$

Where $\alpha_i = 0$ for all objects $\tilde{\mathbf{x}}_i \in \tilde{X}_t$ inside the ellipsoid $\mathcal{E}_{\tilde{M},\mathbf{0}}$ and $\alpha_i \neq 0$ for objects on the surface of the ellipsoid. Only objects for which $\alpha_i \neq 0$ determine the ellipsoid.
The centre $\mathbf{c}$ and the shape matrix E of ellipsoid $\mathcal{E}_{E,\mathbf{c}}$ is computed from this sparse solution:

$$\begin{aligned} \tilde{M} &= \begin{pmatrix} s & \mathbf{v}^T \\ \mathbf{v} & H \end{pmatrix} = \sum_{i=1}^{n} \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T \\ E &= (\delta^{-1} H)^T (\delta^{-1} H) \quad \text{and} \quad \mathbf{c} = -H^{-1}\mathbf{v}(E)^{-1/2} \end{aligned} \tag{3.13}$$

Next, we use the `MVEE` to describe a target class in one-class classification problems.

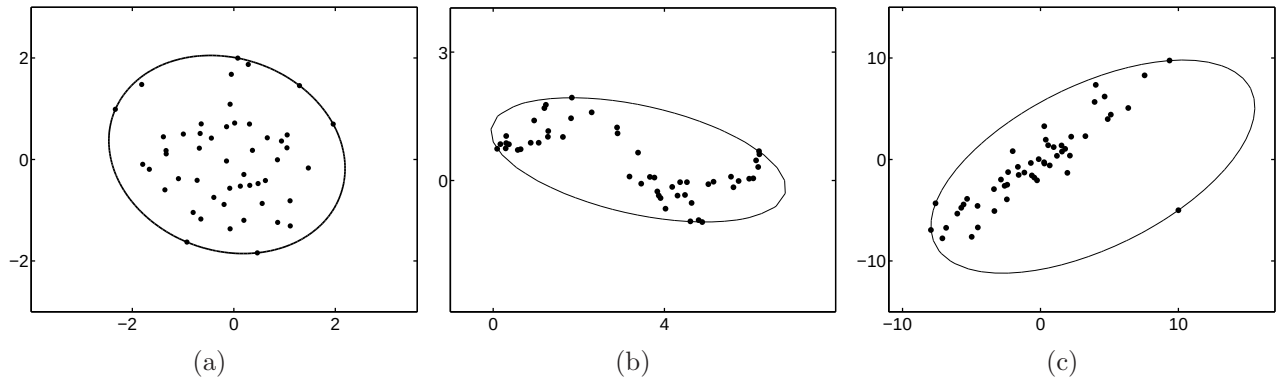


Figure 3.3: Examples of the MVEE for three datasets.

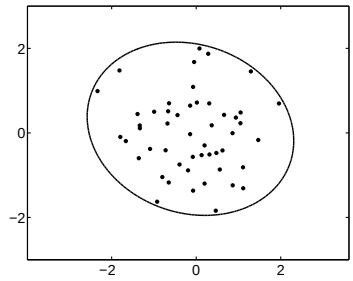


Figure 3.4: MVEE data description.

Figure 3.3 shows three examples of the minimum volume ellipsoid estimated on $2D$ toy problems. We can see that for the first problem, figure 3.3(a), the MVEE model fits well the given data. However, because at least $N+1$ training objects need to be on the surface of the MVEE, we expect a large target rejection rate. Therefore, we change the threshold from 1 to $1 + (\mathbf{x}_i - \mathbf{c})^T E(\mathbf{x}_i - \mathbf{c})$, where $\mathbf{x}_i$ is the closest object, inside the MVEE, to the surface of the MVEE; see figure 3.4.

For the problem in figure 3.3(b), a more flexible descriptor is needed, e.g. the minimum spanning tree data descriptor described in next section. However, if we fit the ellipsoidal description to the data, there is going to be either a large part of a genuine target class outside the descriptor or large empty region inside the descriptor. Such empty region might be an indication of a region in a representation space where a non-target class is located.

In figure 3.3(c) the data distribution has an ellipsoidal shape, however the estimation of parameters, E and $\mathbf{c}$, of the ellipsoid is largely influenced by a single object. We can pose the following question whether this object is a genuine target object or atypical, therefore an outlier object in the training set? In the next section, we change optimisation (3.6) into a more robust estimator of the MVEE, so that the parameters E and $\mathbf{c}$ are only dependent on the bulk of data.

3.2 Robust estimation of the minimum volume enclosing *N*-ellipsoid

We have noticed that in figure 3.3(c) although the bulk of data has an ellipsoidal shape the MVEE is mostly determined by a single object which is remote from the bulk of the data. We can pose two hypotheses: either the object is a genuine target object and the large empty region inside the ellipsoid is due to sampling of the target class or since the object is remote from the bulk of the data we treat it as an outlier, i.e. an atypical target object.

To determine the bulk of the data and a set of potential outliers in the training set we assign a slack variable $\xi_i \geq 0$ to each object from the training set X_t. During the optimisation

of the MVEE we add the sum of slacks to the determinant of the matrix $\tilde{M}$. Additionally, a parameter $C \geq 0$ is introduced, to determine the trade-off between the volume of $\mathcal{E}$ and the sum of slacks. The value of C is crucial, it indicates whether we focus more on the minimisation of the volume of the ellipsoid $\mathcal{E}$ or on enclosing a large fraction of the data. The optimisation (3.6) can be now written as:

$$\min_{\tilde{M},\xi_i} \quad -\ln\det(\tilde{M}) + C\sum_{i=1}^{n}\xi_i, \tag{3.14a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T\tilde{M}\tilde{\mathbf{x}}_i \leq 1+\xi_i, \quad \forall i=1,\ldots,n, \tag{3.14b}$$

$$\tilde{M} \succ 0,\ \xi_i \geq 0, \quad \forall i=1,\ldots,n. \tag{3.14c}$$

Setting the parameter C is not straightforward, there is no natural indication of its value. However, we can notice that the optimisation of (3.14) resemble the optimisation of SVM and as in ν-SVM [Schölkopf et al., 2000b] we can modify the optimisation such that the parameter becomes easier to set. By using a similar trick as [Schölkopf et al., 2000b] we modify the optimisation (3.14) into:

$$\min_{\tilde{M},\xi_i} \quad -\ln\det(\tilde{M}) + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \nu\rho, \tag{3.15a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T\tilde{M}\tilde{\mathbf{x}}_i \leq \rho+\xi_i, \quad \forall i=1,\ldots,n, \tag{3.15b}$$

$$\tilde{M} \succ 0,\ \xi_i \geq 0,\ \rho \geq 0, \quad \forall i=1,\ldots,n. \tag{3.15c}$$

where ν is now a user specified parameter that equals the fraction of objects outside the optimised ellipsoid $\mathcal{E}_{E,\mathbf{c}}$.

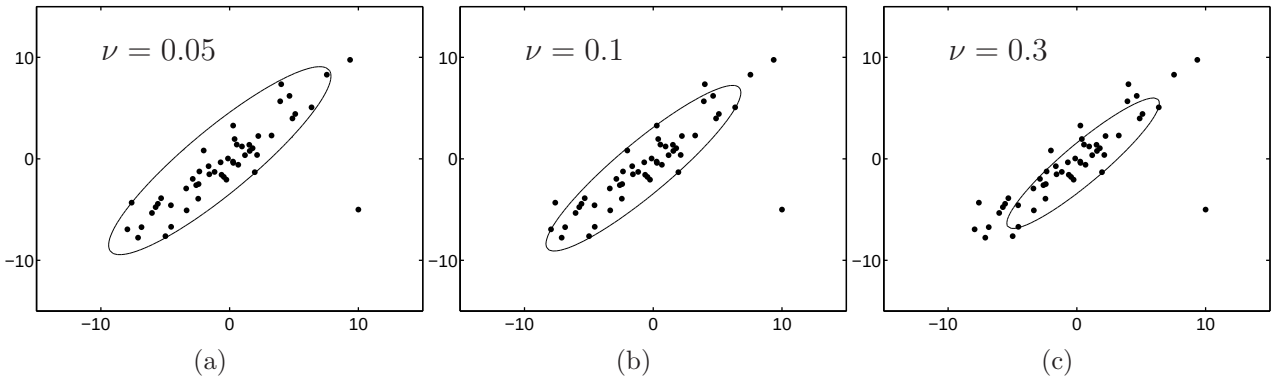


Figure 3.5: Robust MVEE computed by (3.14) with different value of trade-off parameter ν.

The dual of the minimisation is (see appendix A.3 for derivation):

$$\max_{\alpha_i} \quad \ln\det \sum_{i=1}^{n} \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T, \tag{3.16a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i = \nu, \tag{3.16b}$$

$$0 \leq \alpha_i \leq \frac{1}{n}, \quad \forall i = 1, \ldots, n. \tag{3.16c}$$

ρ^* and slack variables ξ_i^* can be computed from the Kuhn-Tucker conditions for the optimal solutions (denoted by *):

$$\begin{aligned} &\alpha_i^*(\rho^* + \xi_i^* - \tilde{\mathbf{x}}_i^T \tilde{M}^* \tilde{\mathbf{x}}_i) = 0, \quad \forall(\alpha_i^* = \frac{1}{n}) \\ &\text{where} \quad \rho^* = \tilde{\mathbf{x}}_i^T \tilde{M}^* \tilde{\mathbf{x}}_i - \xi_i^*, \quad \forall(0 \leq \alpha_i^* < \frac{1}{n}) \end{aligned} \tag{3.17a}$$

$$(\frac{1}{n} - \alpha_i^*)\xi_i^* = 0, \quad \forall(0 \leq \alpha_i^* < \frac{1}{n}) \tag{3.17b}$$

By introducing ξ_i and ν we allow that the fraction ν of objects from the target class is outside the ellipsoid $\mathcal{E}$. Figure 3.5 shows the same data as in figure 3.3(c), however now the ellipsoid is optimised only on a certain fraction of the target class. Objects outside the description are determined by the optimised weights α_i. Training objects inside an ellipsoid $\mathcal{E}_{E,\mathbf{c}}$ have $\alpha_i^* = 0$, objects on the surface the ellipsoid $0 < \alpha_i^* < \frac{1}{n}$, and objects outside the ellipsoid $\alpha_i^* = \frac{1}{n}$.

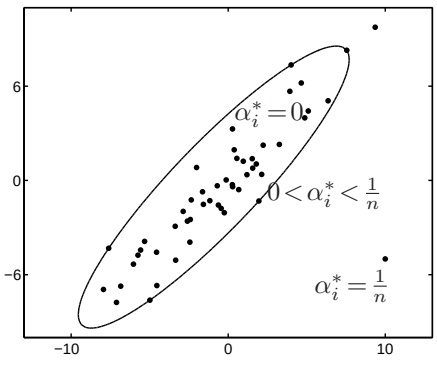


Figure 3.6: Robust `MVEE` with $\nu = 0.05$

Comparing the presented estimator to the existing robust solutions e.g. the minimum covariance determinant [Rousseeuw and van Driessen, 1999] the presented approach is posed as a conic problem therefore it does not need reversal recomputations as existing methods.

3.3 Estimation of the minimum volume enclosing *N*-ellipsoid in the presence of an outlier class

In this section we investigate the algorithm to determine the ellipsoid, that describes the target class, when a few labelled, outlier objects are available during training. There are several ways in which we can formulate the optimisation problem. For example, giving n target objects and m outlier objects, $(k = n+m)$, one can demand that the optimised ellipsoid has all target

objects inside and the sum of distances from the misclassified outlier objects to the surface of the ellipsoid is minimal. This optimisation problem can be formulated as follows:

$$\min_{\xi_i} \quad -\sum_{j=1}^{m} \xi_j, \tag{3.18a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T \tilde{M} \tilde{\mathbf{x}}_i \leq 1, \quad \forall i = 1, \ldots, n, \tag{3.18b}$$

$$\tilde{\mathbf{x}}_j^T \tilde{M} \tilde{\mathbf{x}}_j \geq 1 - \xi_j, \quad \forall j = 1, \ldots, m, \tag{3.18c}$$

$$\tilde{M} \succ 0, \; \xi_j \geq 0, \quad \forall j = 1, \ldots, m. \tag{3.18d}$$

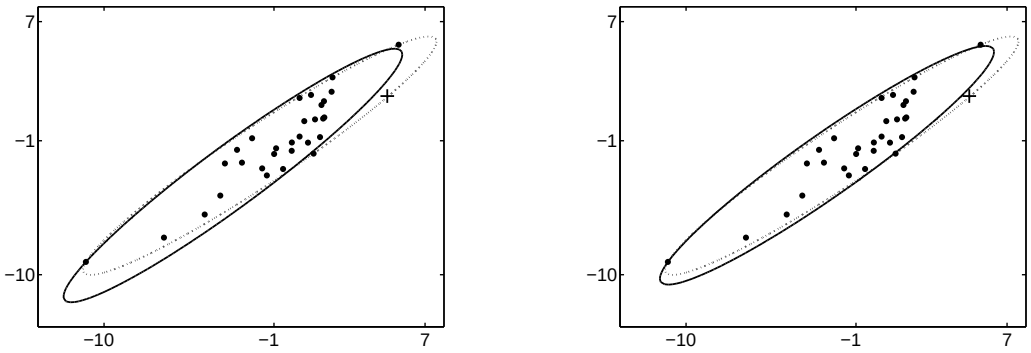


Figure 3.7: `MVEE` computed by (3.18) (a) and (3.19) (b), denoted by the continues line. The doted line denotes `MVEE` computed by (3.12). The target objects are marked $\{\bullet\}$ and a single outlier object by $\{+\}$.

The example of an ellipsoid given by the optimisation 3.18 is shown in the left figure 3.7, it is denoted by the continuous line. For comparison we also plot the ellipsoid given by the optimisation (3.12). The target class is denoted as $\{\bullet\}$ and a single outlier object by $\{+\}$. It should be noticed that (3.18) does not guarantee small volume of the ellipsoid. Since we minimise sum of slacks, there is not a unique solution to the optimisation. However, by adding an additional constraint that the ellipsoid should have a minimum volume a unique solution can be obtained.

$$\min_{\tilde{M}, \xi_i, \xi_j} \quad -\ln\det(\tilde{M}) + C_1 \sum_{i=1}^{n} \xi_i + C_2 \sum_{j=1}^{m} \xi_j, \tag{3.19a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T \tilde{M} \tilde{\mathbf{x}}_i \leq 1 + \xi_i, \quad \forall i = 1, \ldots, n, \tag{3.19b}$$

$$\tilde{\mathbf{x}}_j^T \tilde{M} \tilde{\mathbf{x}}_j \geq 1 - \xi_j, \quad \forall j = 1, \ldots, m, \tag{3.19c}$$

$$\tilde{M} \succ 0, \; \xi_i, \xi_j \geq 0, \quad \forall (i, j). \tag{3.19d}$$

where C_1 and C_2 are two trade-off parameters that allow to treat errors on the target and outlier classes differently. Similarly to the optimisation problem (3.15) we can rewrite (3.19) into:

$$\min_{\tilde{M},\xi_i} \quad -\ln\det(\tilde{M}) + \frac{1}{k}\sum_{i=1}^{k}\xi_i + \nu\rho, \tag{3.20a}$$

$$\text{s.t.} \qquad \omega_i\tilde{\mathbf{x}}_i^T\tilde{M}\tilde{\mathbf{x}}_i \leq \omega_i\rho + \xi_i, \quad \forall i = 1,\ldots,k, \tag{3.20b}$$

$$\tilde{M} \succ 0,\ \xi_i, \nu \geq 0,\ \omega_i \in \{1,-1\}. \tag{3.20c}$$

which simplifies the dual. ω_i denotes the label of object $\mathbf{x}_i \in X_t$, i.e. $\omega_i = 1$ for the target objects and $\omega_i = -1$ for the outlier objects. $k = m + n$ and ν is a user specified parameter indicating the fraction of allowed misclassifications from both classes.
The dual to minimisation (3.20) is (see appendix A.4 for a derivation):

$$\max_{\alpha_i} \quad \ln\det\sum_{i=1}^{k}\omega_i\alpha_i\mathbf{x}_i\mathbf{x}_i^T \tag{3.21a}$$

$$\text{s.t.} \quad \sum_{i=1}^{k}\omega_i\alpha_i = \nu, \tag{3.21b}$$

$$0 \leq \alpha_i \leq \frac{1}{k}, \quad \forall i = 1,\ldots,k. \tag{3.21c}$$

ρ^* and ξ_i^* can be computed from Kuhn-Tucker conditions of the optimal solution:

$$\alpha_i^*(\omega_i\rho^* + \xi_i^* - \omega_i\mathbf{x}_i^T\tilde{M}^*\mathbf{x}_i) = 0, \qquad \forall(\alpha_i^* = \frac{1}{k}), \tag{3.22a}$$

$$(\frac{1}{k} - \alpha_i^*)\xi_i^* = 0, \qquad \forall(0 \leq \alpha_i^* \leq \frac{1}{k}). \tag{3.22b}$$

The optimisation (3.20) does not only minimise the slacks ξ_i but also the volume of the ellipsoid $\mathcal{E}$. Therefore given ν the solution is unique. An example of an ellipsoid obtained by (3.21) is shown in figure 3.7(b). We can see that volume of the obtained ellipsoid is smaller than the ellipsoid in figure 3.7(a). The volume of the ellipsoid optimised by (3.19) is also smaller than the volume of the `MVEE`, however one of target objects is misclassified by the optimised ellipsoid in figure 3.7(b).

3.4 Experiments

In this section we compare the performance of the minimum volume enclosing ellipsoid data description with several similar parametric models: minimum volume enclosing box (`MVEB`), minimum volume enclosing sphere (`MVES`) and a single Gaussian description. The problems (3.12), (3.16) and (3.21) are optimised using `YALMIP` [Löfberg, 2004] and `SeDuMi` [Sturm, 1999] optimisation packages.

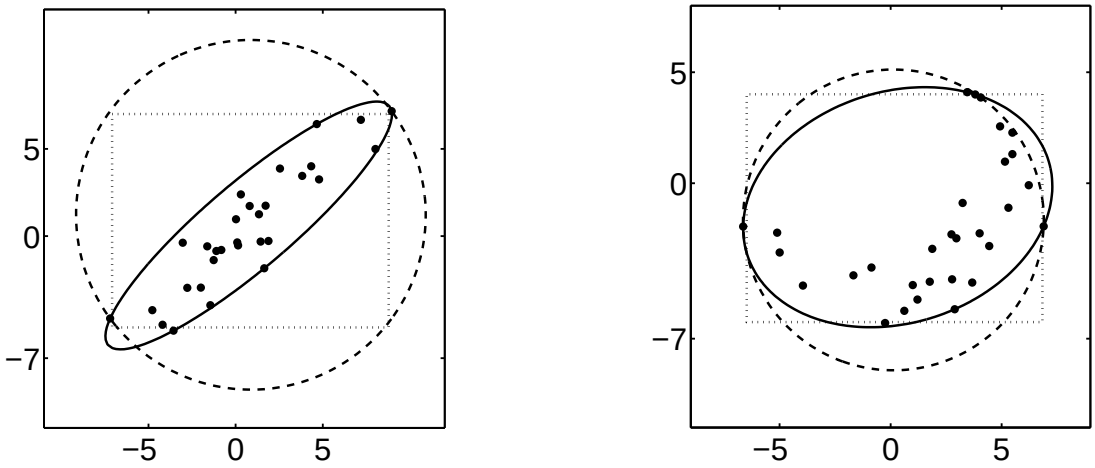


Figure 3.8: The minimum volume enclosing box (dotted line), sphere (dashed line), ellipsoid (continuous line), trained on two datasets.

First we compare the `MVEE` to the minimum volume box and the minimum volume N-sphere. The minimum volume enclosing box is defined as a set of $2N$ hyperplanes perpendicular to coordinate axes:

$$\mathcal{I}\{\min_{\mathbf{x}_i \in X_t} x_{i_j} \leq x_j \leq \max_{\mathbf{x}_i \in X_t} x_{i_j}\}, \quad \forall j = 1, \ldots, N \tag{3.23}$$

The output of the classifier is defined as a distance to the closest hyperplane. The smallest enclosing N-sphere can be computed by (2.9) using a linear kernel $\mathbf{x}_i^T \mathbf{x}_j$. The number of parameters to be estimated is $N + 1$. Finally, the number of parameters in the `MVEE` is $N^2 + N$. Examples of the decision boundaries computed by these three classifiers are shown in figure 3.8. The classifiers are computed with the threshold $\theta = 0.05$.

In tables 3.1 and 3.2 the volume of the classifiers and the classification error, Λ, for each of the three classifiers are shown. The classifiers were trained on *letter* and *mfeat-kar* datasets [Hettich et al., 1998]. Each of the classes in the datasets was set to be a target class and all other classes to be an outlier class. The *letter* dataset is a 16 dimensional problem with about 750 objects per class. The dataset contains the 26 capital latin letters. The *mfeat-kar* dataset is a 64 dimensional problem with 200 objects per class. The classes consist of ten arabic digits. The target class was split half-half on training and test set and the classifiers were only trained on a target class. The volume of classifiers was computed for the threshold set to accept all training objects.

From the tables it can be seen that for the two recognition problems, *letters* and *digits*, the ellipsoidal description has the smallest classification error. This indicates that such description fits the problem well. Moreover, the volume of the ellipsoids is several orders smaller than volumes of `MVEB` and `MVES`. This also indicates good fit on the target class. A descriptor with a small volume might reject outlier objects that were not represented in the test set, non-digit or letter classes. Although one might argue that the `MVEE` is the most complex of three classifiers and the computationally most expensive to train, the `MVEE` required only to compute max and min feature values, `MVES` can be computed by linear programming, to compute `MVEE` we need to use conic programming.

We also compare the performance of the `MVEE` with a single Gaussian description.

Table 3.1: Results for four classifiers trained on each of UCI *letter* dataset's classes. V_B, V_S, V_G and $V_\mathcal{E}$ denote volumes of `MVEB`, `MVES`, a single Gauss and `MVEE`. The volumes in the table should be multiply by factors from the first row. Λ_B, Λ_S, Λ_G and $\Lambda_\mathcal{E}$ are respective mean errors of the classifiers on a test set.

target class	$V_B \times 10^{14}$	$V_S \times 10^{16}$	$V_G \times 10^{14}$	$V_\mathcal{E} \times 10^{10}$	Λ_B	Λ_S	Λ_G	$\Lambda_\mathcal{E}$
A	15.0(1.9)	8.2(2.3)	1.4(1.7)	11.3(2.7)	41.1(1.1)	34.7(0.9)	8.4(3.1)	6.9(1.0)
B	0.9(0.3)	11.5(5.9)	1.0(0.5)	1.9(0.4)	43.5(0.6)	39.4(1.4)	11.5(2.0)	8.5(1.1)
C	3.9(1.0)	2.5(0.8)	0.3(0.1)	4.4(0.5)	43.7(0.5)	35.4(2.5)	13.0(1.8)	9.5(0.9)
D	5.9(1.3)	4.4(0.7)	0.6(0.3)	4.8(0.8)	42.8(0.5)	40.5(0.4)	14.4(1.4)	8.6(1.0)
E	3.9(1.2)	1.7(0.2)	0.1(0.1)	4.4(0.5)	42.1(0.7)	33.8(1.3)	14.1(2.3)	9.6(1.2)
F	7.2(1.8)	7.1(2.1)	0.9(1.2)	6.8(1.5)	44.1(0.4)	40.3(1.9)	16.8(4.0)	9.4(1.3)
G	4.4(1.2)	3.7(1.1)	0.3(0.10)	9.6(1.6)	42.4(0.4)	35.8(1.1)	16.6(0.7)	9.5(1.2)
H	14.4(3.4)	14.8(2.0)	0.10(0.09)	65.1(10.7)	44.4(0.4)	44.8(0.3)	29.4(2.4)	13.4(1.2)
I	1.5(0.3)	2.5(0.2)	41.3(31.4)	7.6(2.6)	43.9(0.2)	40.1(0.6)	19.8(2.7)	7.6(1.5)
J	20.8(3.8)	8.1(1.7)	0.03(0.03)	14.0(2.2)	45.6(0.4)	38.1(0.8)	10.9(1.0)	9.6(1.3)
K	15.9(3.0)	16.6(5.2)	0.01(0.00)	22.0(2.6)	42.3(0.4)	45.3(0.4)	14.7(1.4)	10.5(0.8)
N	37.2(7.2)	51.6(7.8)	0.8(0.7)	13.8(2.1)	43.1(0.2)	43.6(1.3)	18.0(2.3)	8.8(1.5)
M	275.8(69.7)	161.4(66.8)	0.04(0.05)	252.7(37.0)	34.0(1.4)	40.2(0.9)	26.8(3.8)	8.2(1.2)
L	41.3(14.5)	35.3(13.3)	0.10(0.06)	47.3(12.6)	40.6(0.7)	44.2(0.9)	14.5(1.9)	7.6(1.2)
O	1.6(0.5)	4.5(1.6)	13.7(10.1)	3.0(0.8)	42.9(0.2)	36.2(0.8)	14.8(1.9)	7.7(1.1)
P	8.4(2.0)	6.0(0.9)	0.3(0.6)	11.1(2.0)	42.3(1.5)	39.9(0.4)	17.4(5.0)	9.3(1.1)
Q	31.6(6.4)	20.2(5.3)	0.6(0.6)	20.8(2.3)	40.0(0.3)	41.8(1.3)	20.8(2.4)	8.5(0.8)
R	1.8(0.4)	8.8(1.8)	0.2(0.1)	3.8(0.5)	43.4(0.8)	39.9(0.9)	11.7(1.2)	9.2(0.9)
S	12.1(2.3)	9.2(1.3)	0.06(0.06)	21.8(6.0)	43.3(0.2)	38.4(0.5)	20.3(3.2)	9.8(1.6)
T	7.5(0.9)	7.4(0.4)	0.6(0.4)	9.6(2.2)	44.1(0.5)	41.5(0.5)	17.3(2.1)	7.5(0.9)
U	18.2(2.5)	16.0(3.0)	0.1(0.1)	22.5(5.0)	42.6(1.3)	41.9(0.7)	10.8(1.9)	7.2(1.3)
V	3.9(0.6)	4.5(0.5)	0.5(0.4)	4.8(1.1)	42.9(1.0)	37.5(0.4)	5.8(0.6)	7.6(1.4)
W	11.0(4.0)	85.2(42.3)	0.2(0.09)	21.0(6.7)	41.0(1.9)	41.7(3.6)	7.7(0.7)	6.6(1.5)
X	3.5(0.9)	5.9(1.5)	0.7(1.1)	3.8(0.9)	41.1(0.2)	43.1(0.4)	12.8(2.0)	8.5(1.2)
Y	21.3(3.1)	10.1(4.6)	0.04(0.03)	95.8(10.7)	43.2(0.7)	39.4(1.4)	12.3(1.2)	9.4(0.8)
Z	4.8(2.1)	9.7(1.8)	4.9(7.3)	13.0(5.0)	42.2(0.5)	39.2(1.3)	20.9(6.2)	8.7(1.1)

Table 3.2: Results for four classifiers trained on each of UCI *mfeat-kar* dataset's classes. V_B, V_S, V_G and $V_\mathcal{E}$ denote volumes of `MVEB`, `MVES`, a single Gauss and `MVEE`. The volumes in the table should be multiply by factors from the first row. Λ_B, Λ_S, Λ_G and $\Lambda_\mathcal{E}$ are respective mean errors of the classifiers on a test set.

target class	$V_B \times 10^{55}$	$V_S \times 10^{64}$	$V_G \times 10^{45}$	$V_\mathcal{E} \times 10^{31}$	Λ_B	Λ_S	Λ_G	$\Lambda_\mathcal{E}$
0	10.1(0.6)	0.3(0.1)	758.4(505.2)	6.7(2.6)	39.6(3.9)	49.9(0.2)	10.6(2.9)	7.4(1.7)
1	355.6(22.0)	171.5(39.1)	3.1(2.3)	335.2(90.0)	36.2(3.5)	49.5(0.4)	15.0(2.6)	13.7(4.3)
2	100.9(5.1)	0.5(0.1)	17.3(9.6)	21.6(5.5)	37.3(3.3)	49.8(0.3)	7.5(2.7)	6.8(1.6)
3	732.8(38.5)	3.9(1.2)	0.3(0.1)	$1.7(0.5) \times 10^3$	36.2(2.4)	49.9(0.2)	14.5(2.6)	13.5(3.3)
4	2744.7(118.0)	0.9(0.2)	40.8(17.5)	26.6(9.1)	34.9(2.8)	49.8(0.3)	9.1(2.4)	8.0(1.6)
5	4848.6(252.9)	3.8(0.5)	0.01(0.01)	$3.5(1.0) \times 10^4$	40.8(2.9)	49.4(0.5)	16.7(3.5)	16.5(2.3)
6	371.9(9.1)	0.2(0.04)	18.4(10.3)	97.6(44.1)	34.6(4.1)	49.2(0.4)	12.2(1.6)	8.1(3.5)
7	0.04(0.00)	0.2(0.06)	$8.6.(0.8) \times 10^6$	0.00(0.00)	29.9(3.4)	49.2(0.5)	6.3(3.5)	6.3(1.7)
8	$1.5(0.6) \times 10^4$	1.0(0.3)	0.01(0.00)	$1.9(0.4) \times 10^5$	43.8(2.1)	49.9(0.2)	15.1(2.5)	14.7(1.7)
9	4.0(0.2)	0.4(0.08)	52.7(31.2)	14.9(3.4)	37.9(3.4)	49.6(0.4)	11.3(2.6)	9.3(1.9)

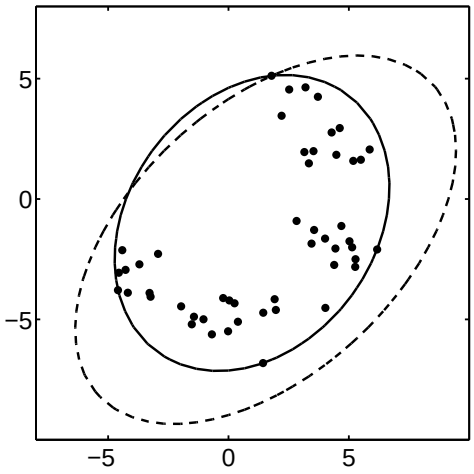


Figure 3.9: The single Gaussian data description (dashed line) and the minimum volume enclosing ellipsoid data description (solid line).

Both classifiers are based on ellipsoids. Although a Gaussian is based on a covariance matrix of the objects and the MVEE on the objects on the surface of an ellipsoid. The difference is clearly visible in figure 3.9. The volume of a Gaussian data description is larger than the volume of the MVEE. Therefore, in general we expect a larger outlier acceptance rate, this can be an undesirable behaviour in applications where robust statistics are required. Moreover, the Gaussian description is sensitive to the type of sampling. If the probability density of a training set does not represent the probability density of a test set the estimated decision boundary is incorrect. For example, imagine a recognition problem in which one trains a classifier to recognise an apple. One of the features we are using is colour. However, the probability of encountering, e.g. a red apple, is different in different places on the Earth. Therefore, the probability density function differs for different test sets, form different places. However, class domains do not change because the same apple encountered in different places is classified with the same label.

In such recognition problems the MVEE outperforms a single Gaussian description. On the other hand when large and representative training set is available the Gaussian might be a better model to describe such problem.

In tables 3.1 and 3.2 we compare the errors and the volumes of the two descriptors. The volume of a Gaussian, with a threshold θ on the pdf of the target class and a covariance matrix Σ, is computed from equation (3.2). The volume of an ellipsoid with the shape matrix E given by (see appendix A.1 for derivation):

$$E = \frac{\Sigma^{-1}}{-2\ln(\sqrt{(2\pi)^N \theta \det(\Sigma)})} \tag{3.24}$$

From the tables we can see that the volume of a Gaussian description is always larger than the volume of the MVEE. Also on *letter* dataset the MVEE outperform the Gaussian description. However, for high dimensional *mfeat-kar* data the difference in AUCs are not significant. This is because in high dimensions also the Gaussian description is mostly influenced by objects on the surface. If data is normally distributed, almost entire dataset is on the surface of the ellipsoid computed by the Gaussian description.

3.5 Conclusions

We have introduced a new one-class classifier based on the minimum volume enclosing ellipsoid (MVEE) algorithm. The basic MVEE algorithm has been extended to cope with outliers present in the training set. Such an algorithm is similar to robust statistics, as the classifier is estimated on a fraction of objects, that minimises the classifier volume. The estimated fraction of outliers present in the data is set by the classifier parameter ν.

In some recognition problems labelled outlier objects might be available during training. The third proposed algorithm based on MVEE takes to account also labelled outlier objects. The classifier has the trade-off parameter between its volume and the fraction of misclassified target and outlier objects.

The proposed classifiers are examples of domain-based one-class classifiers, therefore they suffer less from different sampling methods. The classifiers can achieve good performance on the training data that it is not sampled independently and identically from the class distribution.

We have compared the MVEE with other parametric one-class classifiers: the minimum volume enclosing box, the minimum volume enclosing sphere and the single Gaussian data description. In recognition problems where data is normally distributed but has not been sampled according to its distribution the presented algorithms outperform the single Gaussian and other parametric models. However, when normally distributed target class has been sampled well the single Gaussian and the MVEE have similar performance.

As computation of the MVEE is based on outer products between objects. The kernel trick, as based on inner product relations between objects, can not be used. However, we can "kernelise" the MVEE algorithm by first mapping the data to a Hilbert space and then mapping it to lower dimensional space by the kernel PCA. The MVEE can be trained on such data giving a non-ellipsoidal shape of a decision boundary.

Chapter 4

Minimum spanning tree data description

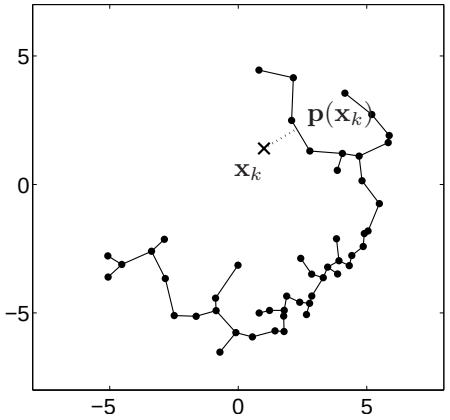


Figure 4.1: The distance of a new object $\mathbf{x}_k$ to a `MST_DD` is measured as the smallest distance between $\mathbf{x}_k$ and its projection $\mathbf{p}(\mathbf{x}_k)$ on an edge of a graph.

In this section a new one-class classifier is proposed. To describe the target class, the classifier describes the training data by a graph, in particular, a Minimum Spanning Tree (`MST`). The classifier is density independent therefore it is an example of a domain-driven one-class classifier.
Our goal is to show that although several one-class classifiers have been proposed, as has been discussed in the last section, many of them are based on similar assumptions. Therefore, the type of recognition problems that can be handled is also similar.
Most one-class classifiers presented in the previous section minimise the spherical volume of a region that captures the target class, e.g. `SVDD` or estimates parameters of an assumed model e.g. $\boldsymbol{\mu}$ and Σ of a Gaussian distribution.

This means by assuming some model, e.g. a sphere or a Gaussian density, one fits the model as good as possible to the given target data. On the other hand classifiers like Parzen or nearest neighbour are nonparametric models, they can model any type of distribution, however to do so they require a large amount of data, especially in high dimensional spaces.

Here, we propose a non-parametric classifier which is based on a graph representation of the training data. The graph is trained to capture the underlining structure of the data. The basic elements of this classifier are not only vertices but also the edges of the graph. This gives a much richer representation of the data. By considering edges of a graph as objects from the target class additional virtual target objects are generated. If our assumptions about the data are correct, this can help to model a target distribution in high dimensional spaces for small sample size problems.

[Li and Lu, 1999] introduced a similar classifier, called the Nearest Feature Line Method (`NFLM`). In the `NFLM` one describes a training set by a set of lines between all pairs of objects from particular class. The new object is classified to one the classes from the training set.

The classification is based on the distance to the nearest line, from the set. In the paper, it has been shown that the proposed classifier performs well in face recognition problems in comparison to nearest neighbour methods. However, the NKLM method has several drawbacks. A test object can be close to a line determined by two training objects, but far from any object in the training set, therefore the method is sensitive to outliers. Moreover, for a large training set the testing stage can be computationally expensive as it is required to compute distance to $\frac{1}{2}\sum_{j=1}^{C} n_j(n_j-1)$ lines for each test object. Where n_j is a number of objects in a class j and C is the number of classes.

The idea of describing a dataset by a graph is also not entirely new, during training of a self-organising map one usually fits a $2D$ or $3D$ grid to represent an underling distribution of the data. However, the result of the training of a SOM is a set of vertices and the edges of the graph are neglected. Moreover, the size and the dimensionality of the grid should be specified beforehand. Here, we would like to describe the target data using the entire graph, including the edges. Moreover, we are interested in a graph description where the structure of the graph is learned from the data and not specified beforehand.

First, we assume that the target data is distributed in a high dimensional space along a non-smooth manifold; see figure 4.1. As has been shown by many researches such problems exist usually when data is represented by row measurements e.g. pixels in hand written characters, faces, gesture recognition. Since some pixels (features) have small variance, e.g. corner pixels, information about the data is usually distributed on a lower dimensional manifold than size of an image. Most of the existing one-class classifiers fail to describe such data correctly as they are based on a Gaussian or a spherical distribution. They include large regions of the space perpendicular to the data manifold. This may cause a large outlier acceptance rate.

We could describe the data manifold by non-parametric density estimation e.g. Parzen or nearest neighbour. However, the finite training set size makes it usually difficult to accurately approximate the density of a target class by non-parametric models. An other possibility would be to use subspace methods e.g. kernel PCA [Schölkopf et al., 1998]. However parameters of such description (parameters of a kernel as well as intrinsic dimensionality of data) have to be estimated from the given data or to be set by a user.

Also several techniques that are used for the visualisation of high dimensional data could be used here. For example, techniques like: isomap [Tenenbaum et al., 2000], local linear embedding [Roweis and Saul, 2000] or multi-dimensional scaling [Gower, 1986]. However, it is difficult to apply such algorithms to new data without recomputing the model itself. In addition such algorithms required several parameters to be specified beforehand. Moreover, the volume of these descriptions are infinite, this again might cause a high outlier acceptance rate.

Because we compute a connected graph, our second assumption about data is that the target class is at most unimodal. Therefore, we can assume that there is at least a single path, which we denote as $\mathbf{x}_i \xrightarrow{\text{path}} \mathbf{x}_j$, between two objects $\{\mathbf{x}_i, \mathbf{x}_j\} \in X_t$ such that each object on that a path also belongs to the target class.

4.1 Minimum Spanning Tree

Since we assumed a small sample size problem, we are interested in a simple graph description, with as few parameters as possible. Our choice is to use the Minimum Spanning Tree (MST) as a graph descriptor of the target class. MST does not have any parameter to set and it is sufficiently flexible to follow a difficult shape of the target class of a non smooth manifold. Moreover, as

during estimation of MST one minimises the total sum of lengths of edges, the estimation of the MST can be related to the idea of the maximum likelihood model selection or the minimum description length. Therefore, for any pair $\{\mathbf{x}_i, \mathbf{x}_j\} \in X_t$ we estimate a single path in a graph. Given the training set X_t by estimating MST one minimises the volume of the target class descriptor and maximises the probability that objects in the neighbourhood of the estimated edges of the graph also belong to the target class. In addition we are looking for the graph with the minimum sum of length of edges which fulfill our assumption about data this can be related to the minimum description length criterion [Kolmogorov and Tikhomirov, 1961].

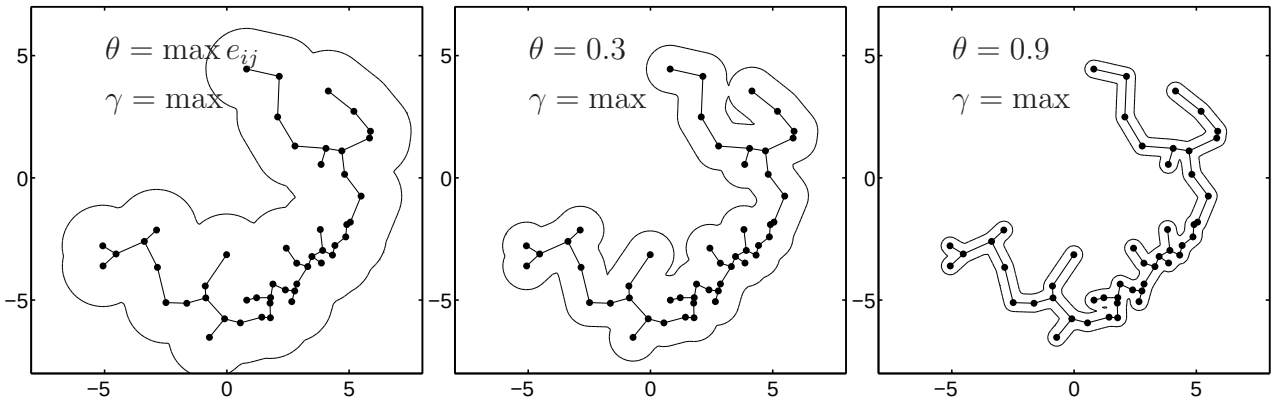


Figure 4.2: Examples of solution given by MST_DD for different thresholds θ. The complexity γ is fixed, the entire MST is considered $\gamma = \max$.

Training the classifier can be formulated as solving the standard MST problem [1]. We are given a $N \times n$ data matrix X_t, from which an $n \times n$ adjacency matrix A is computed. The elements of the adjacency matrix are Euclidian distances between objects $e_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$. The path between two objects $\{\mathbf{x}_i, \mathbf{x}_j\} \in X_t$ is defined as a set of edges from the adjacency matrix:

$$(\mathbf{x}_i \xrightarrow{\text{path}} \mathbf{x}_j) \equiv \{\{e_{kl}^q\}_{q=1}^p : e_{kl}^q \in A, \quad 1 \leq k, l \leq n\} \tag{4.1a}$$

$$\text{where} \quad e_{i\cdot}^1, e_{\cdot j}^p \quad \text{and} \quad e_{\cdot l}^q, e_{l\cdot}^{q+1} \; \forall_{1 \leq q \leq p-1,\, 1 \leq l \leq n} \tag{4.1b}$$

where $\cdot$ denotes any index value. Since we would like to compute a connected graph without loops, this can be formulated as the problem of finding $n-1$ edges with the following properties:

$$\min \quad \sum_{i \neq j}^{n-1} (\mathbf{x}_i \xrightarrow{\text{path}} \mathbf{x}_j) \tag{4.2a}$$

$$\text{s.t} \quad \exists!(\mathbf{x}_i \xrightarrow{\text{path}} \mathbf{x}_j), \quad \forall \{\mathbf{x}_i, \mathbf{x}_j\} \in X_t, \quad i, j = 1, \ldots, n. \tag{4.2b}$$

As the MST problem rises in many theoretical and practical problems several algorithms have been proposed. The most popular algorithms are the Prim's [Prim, 1957] and Kruskal's [Kruskal, 1956]. The basic structure of the algorithms is presented below. We initial each algorithm with an edge e^* of the minimum length.

[1] [Graham and Hell, 1985] gives a history of the problem, which originates to the work of Czekanowski in 1909.

Prim_MST(X_t)
n vertices $\mathbf{x}_j \in X_t$, $n \times n$ edges $e_{ij} \in A$
initialise MST $\leftarrow$ MST $\cup \{e^*\}$,
$A \leftarrow A \setminus \{e^*\}$
while ($|$MST$| < n-1$)
$e^*_{ij} = \min(e \in A)$
s.t. $\mathbf{x}_i \in$ MST, $\mathbf{x}_j \in X_t$,
MST $\leftarrow$ MST $\cup \{e_{ij}\}$
$A \leftarrow A \setminus \{e_{ij}\}$, $X_t \leftarrow X_t \setminus \{\mathbf{x}_j\}$
end
Algorithm 4.1: Prim's algorithm.

Kruskal_MST(X_t)
n vertices $\mathbf{x}_j \in X_t$, $n \times n$ edges $e_{ij} \in A$
initialise MST $\leftarrow$ MST $\cup \{e^*\}$,
$A \leftarrow A \setminus \{e^*\}$
while ($|$MST$| < n-1$)
$e^*_{ij} = \min(e \in A)$
s.t. $\{\mathbf{x}_i, \mathbf{x}_j\} \notin$ MST,
MST $\leftarrow$ MST $\cup \{e_{ij}\}$
$A \leftarrow A \setminus \{e_{ij}\}$
end
Algorithm 4.2: Kruskal's algorithm

The algorithms differ in that the Prim's algorithm requires that the next edge added be incident with a vertex in the partial tree, MST, whereas the Kruskal's algorithm just adds the next edge that does not form a circuit.

Since the computation complexity of MST is $\mathcal{O}((n-1)^2 \log(n-1))$, it is lower than computation complexity of most machine learning algorithms e.g. SVM $\mathcal{O}(n^3)$. Recently even less computationally expensive algorithms have been proposed [Cormen et al., 1990].

4.2 MST_DD

The minimum spanning tree data description (MST_DD) models a target class by $n-1$ edges of the MST. A new object $\mathbf{x}$ is mapped on the $n-1$ lines determined by n training objects; see figure 4.1. The projection of $\mathbf{x}$ onto a single line determined by $\mathbf{x}_i, \mathbf{x}_j \in X_t$ is computed as:

$$\mathbf{p}_{e_{ij}}(\mathbf{x}) = \left[\left(\frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|} \right)^T (\mathbf{x} - \mathbf{x}_i) \right] \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|} + \mathbf{x}_i \tag{4.3}$$

The distance of a new objects $\mathbf{x}$ to the target class is computed as the minimum distance to the set of $n-1$ edges, therefore either to the line that is determined by each pair in MST or to training objects X_t that define ends of edges.

if $(\min(\mathbf{x}_i, \mathbf{x}_j) < \mathbf{p}_{e_{ij}}(\mathbf{x}) < \max(\mathbf{x}_i, \mathbf{x}_j))$

$$d_{MST\_DD}(\mathbf{x}, X_t) = \min_{e_{ij} \in MST} \min(\|\mathbf{x} - \mathbf{p}_{e_{ij}}(\mathbf{x})\|, \|\mathbf{x} - \mathbf{x}_i\|, \|\mathbf{x} - \mathbf{x}_j\|) \tag{4.4a}$$

else

$$d_{MST\_DD}(\mathbf{x}, X_t) = \min_{e_{ij} \in MST} \min(\|\mathbf{x} - \mathbf{x}_i\|, \|\mathbf{x} - \mathbf{x}_j\|) \tag{4.4b}$$

The decision whether $\mathbf{x}$ belongs to the target or outlier class is based on the threshold set on the distance $d_{MST\_DD}(\mathbf{x}, X_t)$.

$$h_{MST\_DD} \equiv \mathcal{I}(d_{MST\_DD}(\mathbf{x}, X_t) \leq \theta) \tag{4.5}$$

The threshold can be determined based on distances between objects in MST, e.g. the mean length of the edges in the MST. In such a case, MST_DD has not got any parameter to be set or optimised.

In figure 4.2 examples of the MST_DD solutions for a $2D$ toy problem are shown. We can see that the MST classifier describes the data in a different way than standard classifiers shown in figure 2.4. It rather emphasises relations between objects than densities or regions in the target data. Additionally, we also notice from figure 4.2 that the graph can be simplified, by removing some edges without changing the class description causing not connected objects.

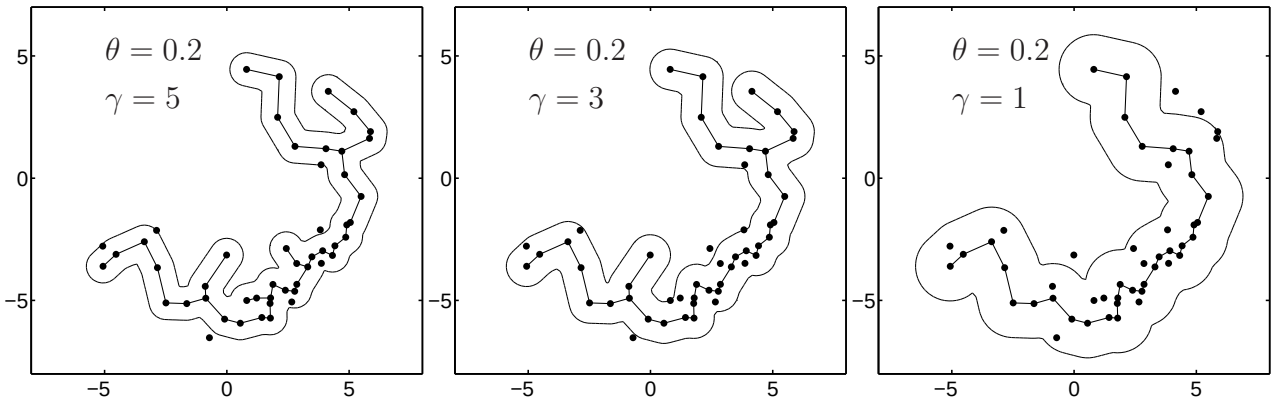


Figure 4.3: Examples of solution given by MST_DD the same threshold θ and different value of the complexity γ.

4.3 Complexity parameter

We define the longest path in the MST as the first principal direction of the graph. The first principle direction is the nonlinear path along which the variance in the training set is maximum. The second longest path, without edges that are part of the first principal direction represents the second principal direction, as so on. Figure on the right shows the first and second principle direction in MST denoted by dotted and dashed lines respectively.
We can simplify the graph representation of data by considering only a number of principle directions. The computation of all paths in the graph can be done using Dijkstra's algorithm, $\mathcal{O}(n^3)$, or the Jordan's algorithm, $\mathcal{O}(n^2 \log(n))$, [Cormen et al., 1990]. To simplify the graph description of the target data we use the Jordan's algorithm since it is more computationally efficient and moreover it allows to compute all paths in MST simultaneously.

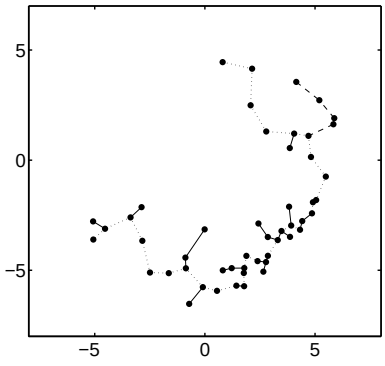


Figure 4.4: First (dotted line) and second (dashed line) principal directions in MST.

From the initial $n \times n$ adjacent matrix we have simplified a description of the target data to $n-1$ edges. In addition, we introduce a complexity parameter γ which specifies how many principle directions should be used to describe the target data. Now, the threshold θ can be set as the error on the fraction of objects, from the training set, that has not been used to describe the target class.

Figure 4.3 shows the MST_DD with a fixed value of the threshold, $\theta = 0.2$, and several values of a complexity parameter γ. We can see that the simplest classifier, $\gamma = 1$, describes the region of the data almost as good as the MST_DD with maximum complexity, equals to the complete MST. This is because the intrinsic dimensionality of the data in figure 4.3 equals one.

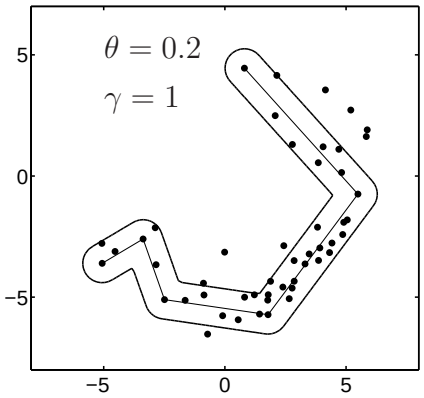


Figure 4.5: Simplified MDS_DD.

Therefore, it is sufficient to use a single principle direction to describe our $2D$ toy problem. We can simplify the classifier even further by removing superfluous vertexes and associated edges. From the existing graph a vertex and two associate edges are removed if a distance between the vertex and its projections onto the new edge is larger than a specified constant. We use equation (4.3) to compute the projection of the vertex to the new edge. If for $\{\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k\} \in X_t$, $\{e_{ik}, e_{kj}\} \in MST$ and $\|\mathbf{x}_k - \mathbf{p}_{e_{ij}}(\mathbf{x}_k)\| \geq \epsilon$ edges e_{ik} and e_{kj} are replaced with an edge e_{ij} in MST_DD. Therefore, the complexity of a classifier can be adjusted to the complexity of the data. The result of such a simplification of an MST_DD is shown in figure 4.5.

Because in one-class classification problems an outlier class might be not available during training, the complexity parameter γ needs to be estimated using some assumptions about the distribution of the outlier class or using some other criterion. This problem is discussed in the next chapter.

4.4 Experiments

In table 4.1 the comparison of the performance between the MST_DD and other one-class classifiers is shown. The comparison is based on the value of AUC. Results for five UCI repository [Hettich et al., 1998] datasets are shown. The name of a target class is indicated in the brackets, after the name of a dataset. The size of the target class, the outlier class and the dimensionality of the problem are stated below the names of the datasets. Complexity parameters for Parzen and kNN were optimised by the leave-one-out error criterion. For k-means, k-centres, MoG and SOM, γ was set to 5. One-class PCA retains 0.95 variance of the training set. For SVDD, MPM and LPDD, $\gamma = 1$. In MST_DD the complete MST is used, $\gamma = \max$. The target class was split half-half between a training and test set and the experiments were repeated 20 times. The threshold was set to 0.1 for all classifiers.

The datasets represent a collection of a small sample size problems, were number of samples is smaller or similar to the number of dimensions. Therefore, we can see that the MST_DD always performs best but not always significantly.

How similar the prediction of MST_DD, is to that of other classifiers can be seen in figure 4.6. The two figures show the characterisation of one-class classifiers by their label disagreements [Duin et al., 2004]. The distance between classifiers is measured on their classification labels. Next, the high dimensional representation is mapped to a $2D$ space using MDS [Cox and Cox, 1994]. The oracle, the classifier that predict true labels, is added as a comparison. The oracle is denoted by O in the figures. From these figures we can see that the MST_DD is most similar, in the sense of predicted labels, to NN, kNN, k-centres and auto-encoder

Table 4.1: The value of AUC with standard deviations (in brackets) for selection of one-class classifiers for small sample size problems.

classifier	AUC				
	nist(0)	vowel(5)	spectf(0)	glass(nonfloat)	sonar(mines)
	200/1800/256	48/480/10	95/254/44	76/9/138	111/60/97
Gauss	97.3(1.3)	93.3 (1.5)	93.3(3.3)	68.0(2.2)	68.0(3.1)
MoG	0(0)	97.6 (1.2)	97.6(3.1)	76.4(2.4)	76.4(3.5)
naïve Parzen	83.6(5.5)	90.2 (1.6)	90.2(3.7)	53.2(2.0)	53.2(3.9)
Parzen	0(0)	97.9 (1.3)	97.9(2.7)	80.5(2.3)	80.5(3.1)
k-means	97.6(0.7)	92.3 (2.2)	92.3(1.7)	69.8(2.5)	69.8(3.7)
NN	86.6(5.7)	92.6 (4.0)	92.6(2.9)	76.3(6.2)	76.3(4.3)
kNN	98.0(0.5)	92.3 (1.7)	92.3(1.5)	69.6(2.9)	69.6(4.8)
auto-encoder	93.2(2.8)	90.7 (1.7)	90.7(6.2)	59.6(3.2)	59.6(6.5)
PCA	98.2(0.8)	90.1 (1.7)	90.1(3.0)	69.6(3.6)	69.6(3.3)
SOM	96.9(0.8)	97.5 (1.1)	97.5(2.1)	80.1(3.0)	80.1(3.4)
MST_DD	98.3(0.6)	98.1 (1.2)	98.1(2.6)	81.1(2.8)	81.1(3.1)
k-centres	96.9(0.7)	90.9 (1.3)	90.9(1.6)	66.8(5.1)	66.8(4.1)
SVDD	0.3(0.2)	97.8 (1.1)	97.8(3.3)	76.1(3.3)	76.1(3.2)
MPM	0.3(0.2)	97.0 (1.2)	98.0(7.4)	78.5(3.9)	78.5(3.0)
LPDD	0.3(0.2)	93.4 (1.5)	93.4(3.3)	63.6(3.6)	63.6(2.7)

and it has large distance to density-based classifiers as well as support vector based classifiers SVDD, MPM. Although the values of AUC for MST_DD, PCA and SOM are quite similar, the predicted labels are different, since the distances, in figures 4.6, between the classifiers are large. Since the distance to the standard one-class classifiers is large the MST_DD might be considered as a valuable member of a committee where one-class classifiers are combined.

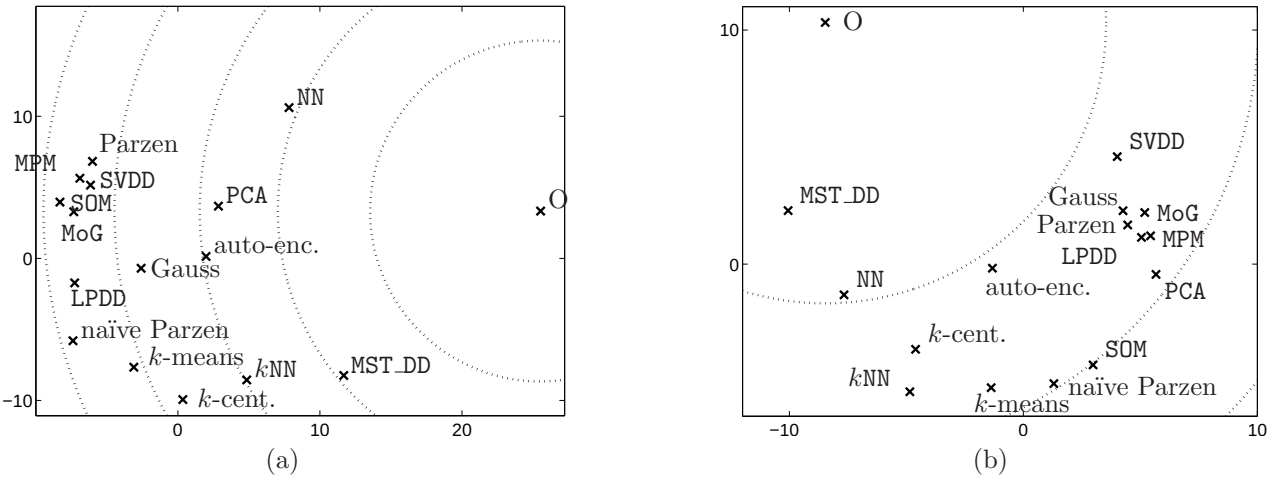


Figure 4.6: The characterisation of classifiers by their disagreement on two datasets vowel: (a) and spectf (b).

4.5 Conclusions

In this section we have proposed a one-class classifier based on the MST. In the simplest version, the classifier does not have any parameters to be optimised. The complexity of the classifier equals the complexity of the MST and the threshold is set as the average length of edges in the MST. As the basic elements of the classifier are edges of the graph additional, virtual training objects are available. Therefore, the presented classifier performs well in high dimensional spaces and small sample size problems in comparison to other one-class classifiers. Since the MST_DD is based on quite different assumptions compared to other one-class classifiers, this makes it an valuable member of a committee in combing classifiers. Possible extensions of MST_DD are the one-class classifiers based on different graph structures e.g. the minimum Stainer tree.

Part II: Model selection in one-class classification

Old Boniface he took his cheer,
Then he bored a hole through a solid sphere,
Clear through the centre, straight and strong,
And the hole was just six inches long.

Now tell me, when the end was gained,
What volume in the sphere remained?
Sounds like I haven't told enough,
But I have, and the answer isn't tough!

Summary of Part II: Model selection in one-class classification

In this part we discuss data-driven methods for the optimisation of parameters of classifiers in the one-class classification problem. Although the performance of classifiers strongly depends on the value of their parameters, often the selection of parameters in a classification or a recognition problem is left to the decision of a person. By comparing several models with user selected parameters, the comparison of performances is made based on rather skills of the person, who selects parameters, and not actually on methods themselves. Here, we focus on automatic, data-driven optimisation of parameters for one-class classifiers. The complexity of a classifier is determined by its parameters/ Moreover, in one-class classifiers, these parameters also determine the trade-off between the error on a target and the error an outlier class. In this part a criterion for selecting complexity parameters for one-class classifiers is proposed. The criterion relies on the minimisation of the ratio of two volumes: the volume of the largest N-sphere [2] containing no training examples, found inside a one-class classifier, divided by the volume of the classifier. By minimising the ratio of these two volumes we minimise the possibility of accepting an outlier object which is remote from given target data and a decision boundary. The ratio is used as a measure of how well a model describes the data. Since a one-class classifier is a bounded subset of a space, its volume is the indicative measure of its performance. In general, a large volume of a classifier indicates a large error on the outlier class and a small error on the target class. On the other hand, a small volume of a classifier indicates a small error on the outlier class and a large error on the target class. A small ratio of these two volumes, indicates a classifier with a large volume, therefore the small error on a target class, and a small volume of empty space, therefore a small error of the outlier class. The proposed method is compared with standard model selection criteria for one-class classifiers: a method based on the generation of uniformly distributed outliers [Tax and Duin, 2001], and a method based on the consistency of a one-class classifier with a given threshold [Tax and Müller, 2004].

To compute the ratio of two volumes several subproblems have to be solved. In section 6.1.1, we present a formula to compute a tight approximation of the volume of one-class classifiers consisting of several intersecting N-spheres e.g. k-means [Jiang et al., 2001], k-centres [Hochbaum and Shmoys, 1985] and self-organising maps (SOM) [Parra et al., 1996]. The proposed approach can tightly approximate the volume of a given one-class classifier in any number of dimensions. In the same section, we derive a formula to compute the volume of a spherical cap in an arbitrary number of dimensions and present a method to check whether more than two N-spheres have a common region. Next, in section 6.1.2, we propose an algorithm to find the largest empty N-sphere in one-class classifiers consisting of N-spheres. Here, we propose a method to check whether an N-sphere is entirely inside a set of intersecting N-spheres. Section 6.1.3 presents an explanation why the presented algorithm does not work for spherical kernel based one-class classifiers such as SVDD [Tax and Duin, 1999] and *oc*-SVM [Schölkopf et al., 2000a]. Finally, an approximately largest N-sphere searching algorithm is presented in section 6.1.4 that is applicable to any one-class classifiers. The proposed algorithms are tested on UCI repository datasets and the presented model selection method is compared with existing methods.

[2] We adapt the notation from geometry where an N-sphere refers to a hypersphere in N dimensions. Therefore the letter N denotes the dimensionality of a input space $\mathbb{R}^N$.

Chapter 5

Model selection methods for one-class classifiers

5.1 Considerations

In the problem of one-class classification, one of the classes, called the target class, has to be distinguished from all other possible objects, called outliers. The goal is to find a boundary descriptor of the target class such that its members are accepted and non-target objects are rejected. It is usually assumed that no outlier examples are available in the training stage, so the classifier should be determined by objects from the target class only. This is an ill-posed problem and it can easily lead to either under- or overfitting as the provided information supports the decision boundary only from one side. How tight a one-class classifier fits the target class is determined by the complexity parameter of the classifier.

The one-class classifier is trained as a boundary descriptor, hence it focuses on finding the boundaries of the target class in the input space. Therefore, when we say that an object is inside a one-class classifier, it means that it lies within its boundaries. Moreover, under the assumption that the outlier class in uniformly distributed [Tax and Duin, 2001], as a one-class classifier is a bounded subset of some space, its volume is an indicative measure of the performance of the classifier. Using a too simple model, figure 5.1(a), yields a large volume of a target class descriptor and results in the acceptance of many outlier examples, giving poor specialisation on the target class. On the other hand a too complex model, 5.1(c), yields a small volume of the target class descriptor and results in the rejection of many target examples, giving poor generalisation on the target class.

Since in the one-class classification problem only one class of data is representative and easily available, the decision boundary is supported only by the target examples. It is, therefore, hard to decide how tightly the boundary should fit around the data in the input space. The lack of outlier examples makes it hard to make a reliable estimation of the error that the classifier makes on novel examples. The error on the target class ε_t, the target rejection rate, can be estimated on the training set. The error on the outlier class ε_o, however, can be only estimated by making additional assumptions concerning the distribution of outlier objects. The most common assumption is that the outlier class is uniformly distributed in the input space [Tax and Duin, 2001, Markou and Singh, 2003b]. The uniform distribution of outliers is also assumed here as the worst case scenario in case of the acceptance of outlier objects.

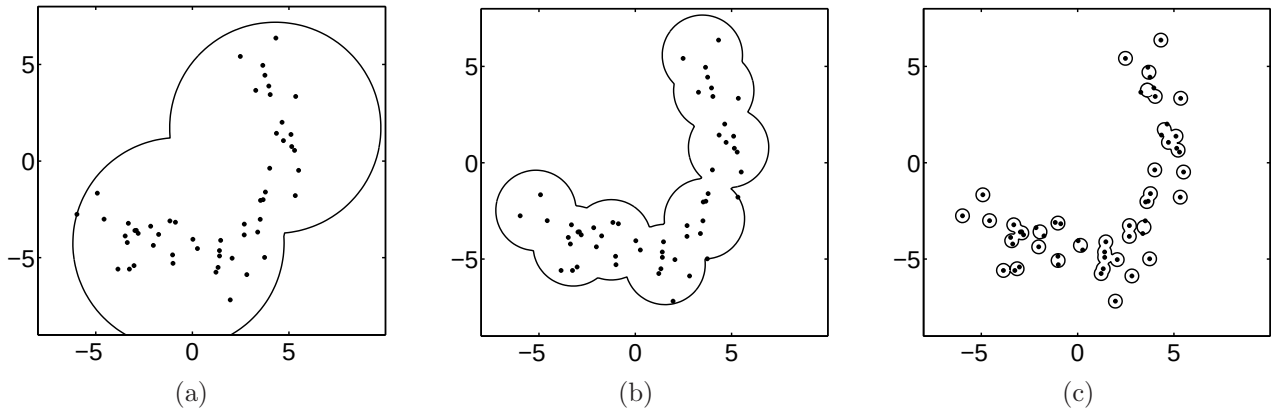


Figure 5.1: 2- 8- and 40-kmeans classifier with $\varepsilon_t = 0.01$.

5.2 Existing model selection methods in one-class classification

Let $X_t = \{\mathbf{x}_i : \mathbf{x}_i \in \mathbb{R}^N, i = 1, \ldots, n\}$ be a training set drawn from the target distribution $p(\mathbf{x})$. Assume a characterisation of this target class by a one-class classifier is sought. In general, one-class classifiers can be presented in the following form:

$$f(\mathbf{x}|X_t, \gamma) = \mathcal{I}(h(\mathbf{x}|X_t, \gamma) < \theta) = \begin{cases} 1, \ \mathbf{x} \text{ is a target,} \\ 0, \ \mathbf{x} \text{ is an outlier,} \end{cases} \tag{5.1}$$

where the function h models the proximity of a vector $\mathbf{x}$ to the training, target data X_t, θ is the specified threshold and $\mathcal{I}(\cdot)$ is the indicator function. Furthermore, γ is a parameter of the model h, indicating its complexity. The threshold θ is optimised to reject a certain, usually user-specified, fraction of the target class ε_t^{tr}, e.g. $\varepsilon_t^{tr} = 0.05$. The ε_t^{tr} has to be determined by the user from a given application, for example by specifying the maximum number of allowed fault alarms in a machine condition monitoring. Apart from the threshold θ the other parameter that determines the performance of a one-class classifier is γ, which denotes the complexity of a chosen model h. Examples of γ are sigma in `SVDD` with a Gaussian kernel, the smoothing parameter in the Parzen density estimation, the number of means or centres in k-means, the mixture of Gaussians and k-centres or the number of hidden units in the auto-encoder. The complexity parameter γ can be determined if it would be possible to determine during training both errors on the target and outlier class e.g. using cross-validation. In general, the error of a one-class classifier can be expressed as:

$$\Lambda(\varepsilon_t, \varepsilon_o) = \lambda\varepsilon_t + (1 - \lambda)\varepsilon_o, \tag{5.2}$$

where λ is a trade-off parameter between the importance of outlier acceptance, ε_o, and target rejection, ε_t. For $\lambda = 0.5$ both errors are equally treated. However, in one-class classification problems during training only examples of the target class are available. Therefore, only ε_t can be estimated. The expected error on the outlier class ε_o only can be estimated by making additional assumptions. Having given estimates of ε_t and ε_o, the complexity of a classifier γ can be optimised. Several techniques have been proposed to optimise γ for one-class classifiers. We can distinguish methods that can be applied to all of one-class

classifiers (e.g. the consistency model selection), methods based on the uniform outlier generation, and classifier specific methods (e.g. the maximum likelihood criterion for density estimators) to estimate the outlier acceptance rate. Next, we describe these methods in details.

Consistency based model selection
In [Tax and Müller, 2004], the authors propose to increase the complexity of a one-class classifier until it becomes inconsistent with the user specified error on the training set ε_t^{tr}. In this approach, no implicit assumption on the outlier distribution is made. The authors assumed that when the complexity parameter γ increases, ε_t also increases. The parameter γ is chosen as the most complex model which is still consistent with a specified ε_t^{tr}. The one-class classifier is considered inconsistent if the estimated error on the target class $\widehat{\varepsilon_t}$ is:

$$\widehat{\varepsilon_t} \geq \varepsilon_t^{tr} + 2\sqrt{\frac{\varepsilon_t^{tr}(1-\varepsilon_t^{tr})}{\mathcal{B}}} \tag{5.3}$$

where $\mathcal{B}$ is the number of binomial experiments. This method does not require knowledge about the distribution of outliers ε_t^{tr} should be specified by a user.

Model selection based on the uniform outliers generation
In [Tax and Duin, 2001] the authors assumed that the outliers are uniformly distributed in the input space. This means that when the chance of accepting an outlier object is minimised, the volume covered by the one-class classifier should be minimised. The authors fit the smallest enclosing N-sphere around a training set X_t, computed by the linear-`SVDD` algorithm [Tax and Duin, 1999], and generate Q uniformly distributed outlier objects inside it. The ratio of generated outlier objects that are classified as targets is the estimation of an outlier acceptance rate ε_o. The authors proposed the following algorithm to generate uniformly distributed outliers in the smallest N-sphere $\mathcal{S}(\mathbf{A}, R)$ that encloses all training objects. Where $\mathbf{A}$ denotes the centre and R radius of an N-sphere.

1. Generate Q points $\mathbf{z}'_i \in \mathbb{R}^N$, from a spherical Gaussian distribution; $\mathbf{z}' \sim \mathcal{N}(\mathbf{0}, I)$.
2. Use a N-dimensional cumulative distribution of χ^2_N, $\mathcal{X}^2_N$, to compute a scaling factor s distributed as $\sim R^N$ on the $[0, 1]$ interval; $s = (\mathcal{X}^2_N(\|\mathbf{z}\|^2))^{\frac{1}{N}}$.
3. For every $\mathbf{z}'_i$, compute $\mathbf{z}_i = \frac{s}{\|\mathbf{z}'_i\|}\mathbf{z}'_i$. These are points uniformly distributed inside the N-sphere $\mathcal{S}(\mathbf{0}, I)$ in $\mathbb{R}^N$.

Algorithm 5.1: Generate Q points uniformly distributed inside an N-sphere $\mathcal{S}(\mathbf{0}, I)$.

The target rejection rate ε_t is estimated using cross-validation on a given target set. The model is selected for which the estimation of the weighted sum of errors in equation (5.2) has the minimum value. This method is accurate up to 15-20 dimensions, depending on the target data distribution, since it requires a large number of artificially generated outliers in high dimensional spaces. The method can be used to estimate the volume of a non-convex body, e.g. the volume of a one-class classifier, that is inside the N-sphere $\mathcal{S}(\mathbf{A}, R)$. The volume of $\mathcal{S}(\mathbf{A}, R)$ multiplied by the estimate of ε_o gives an approximation of the volume of a one-class classifier.

Model selection methods for density based one-class classifiers
There are also model selection methods for specific classifiers, e.g. the maximum likelihood

criterion for density estimators like the Parzen density estimator [Duin, 1976] or the mixture of Gaussians.

Additionally, several heuristics exist that provides an approximate estimate of the complexity parameters γ of the chosen model. For example, the average distance to the $\sqrt{|X_t|}$ nearest neighbour for estimating σ in support vector methods with a Gaussian kernel. Although these methods provide sometimes very good initial guess, they are not very well scientifically founded, therefore they are not considered here.

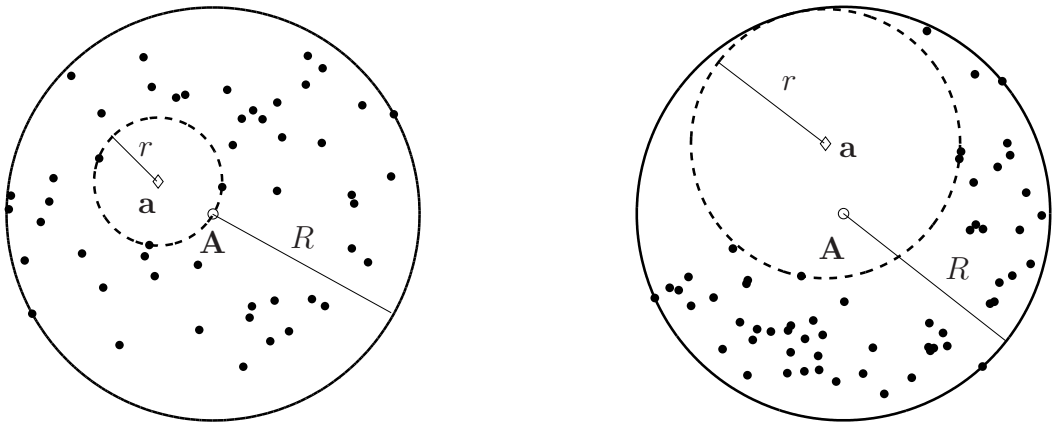
Chapter 6

Volume based model selection in one class classification

Model selection methods rely usually on a statistic computed on a training set. The most common statistic is based on the minimisation of the probability that a test object is misclassified, called the expected classification error or simply the classification error. The expected classification error is estimated using the region of overlap of class posterior probabilities $P(\omega|\mathbf{x})$ estimated by a classifier. In situations, where we can not reliably estimate class posterior probabilities the model selection can be based directly on an error estimate using cross-validation. Other examples of model selection methods assume certain properties of a classification problem. For example, we can assume that a problem is linearly separable and select a model which has the maximum margin between two classes [Vapnik, 1998]. Similarly, for the separable case, one can select a model based on the maximum entropy criterion [Jaakkola et al., 1999]. Since in one-class classification problems only examples of a target class are available, during training, none of the mentioned methods can be used to select an appropriate model without making assumptions about a distribution of an outlier class [Tax and Duin, 2001].

In most outlier detection problems such as: medical screening, machine diagnosis, person identification etc. we are not only interested in classifiers that have a small error but also in what type of objects are misclassified. One can imagine an example where a classifier has a small error but still accepts abnormal objects that are remote from normal target objects. In detection problems where the acceptance of remote outliers is especially dangerous, e.g. machine fault detection we are also interested in the minimisation of acceptance of outlier objects which are far from target objects.
Since we can not estimate the error on the outlier class, in one-class classification problems, without making additional assumptions about the outlier class distribution, we focus on a different goal. Based on the compactness hypothesis [Duin, 1999], we state that empty regions, i.e. regions without training-target objects, inside a one-class classifier are likely to accept outlier objects without significantly improving the generalisation on a target class. Therefore, such regions can cause increase in the outlier acceptance rate without significantly increasing the target acceptance rate. It is assumed that the target class is well sampled. Therefore, we can select models based on the above reasoning. Looking on figure 5.1 one sees that, without outlier objects, or an assumption about the distribution of outlier objects, the decision which regions are superfluous is not unique.

One of the possibilities is to approximate the empty region inside the target region described by the one-class classifier by a convex hull, which does not contain training objects, and it is entirely inside the classifier. However, the computation of an arbitrary convex hull inside a



(a)

Figure 6.1: The largest empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ that can be found inside the one-class classifier based on the smallest N-sphere $\mathcal{S}(\mathbf{A}, R)$ that encloses the target data. The centres of the two spheres are marked by $\circ, \diamond$ respectively. (a): an N-sphere $\mathcal{S}(\mathbf{a}, r)$ is determined by objects from X_t. (b): an N-sphere $\mathcal{S}(\mathbf{a}, r)$ is defined by objects from X_t and the projection of $\mathbf{a}$ onto the surface of a classifier.

classifier is quite cumbersome. Therefore, we chose the most simple, one an N-sphere.

The proposed model selection criterion is to select that model, from a set of possible models, that has the minimum volume of the largest empty N-sphere that can be determined inside the classifier. This criterion can be reformulated as the minimisation of the maximum distance r from the centre of the empty N-sphere $\mathbf{a}$ to an object $\tilde{\mathbf{x}}_i \in \{X_t \cup \mathbf{p}(\mathbf{a})\}$. Where $\mathbf{p}(\mathbf{a})$ is the projection of $\mathbf{a}$ onto the surface of a one-class classifier h.

$$\min \quad r, \tag{6.1a}$$

$$\text{s.t} \quad r = \max_{\mathbf{a}} \min_{i} ||\tilde{\mathbf{x}}_i - \mathbf{a}||, \tag{6.1b}$$

$$\mathcal{I}(h(\mathbf{a}|X_t, \gamma) < \theta) = 1, \tag{6.1c}$$

$$\tilde{\mathbf{x}}_i \in \{X_t \cup \mathbf{p}(\mathbf{a})\},\ \mathbf{a} \in \mathbb{R}^N. \tag{6.1d}$$

Criterion (6.1) selects the model that minimises the largest empty N-sphere inside a one-class classifier. It can be also interpreted as the measure of how well the model fits the target data. We denote the largest empty N-sphere with a centre $\mathbf{a}$ and a radius r as $\mathcal{S}(\mathbf{a}, r)$. The centre $\mathbf{a}$ of the largest empty N-sphere is equidistant to $N+1$ objects in the set $\{X_t \cup \mathbf{p}(\mathbf{a})\}$. The volume of $\mathcal{S}(\mathbf{a}, r)$ is determined as the volume of an N-sphere with radius r in $\mathbb{R}^N$:

$$V_{\mathcal{S}(\mathbf{a},r)} = \frac{2\pi^{N/2} r^N}{\Gamma(N/2+1)}, \qquad \text{with} \qquad \Gamma(x) = \int_0^\infty e^{-t} t^{x-1} dt$$

An example of $\mathcal{S}(\mathbf{a}, r)$ inside a one-class classifier based on the smallest N-sphere enclosing data, denoted as $\mathcal{S}(\mathbf{A}, R)$, is shown in figure 6.1. In figure 6.1(a) $\mathcal{S}(\mathbf{a}, r)$ is determined by objects from X_t only and in figure 6.1(b) by objects in X_t and $\mathbf{p}(\mathbf{a})$.

If we would base the model selection only on criterion (6.1) we are biased towards classifiers with a small volume. In that case, we would select classifier (c) in figure 5.1. On the other hand the volume of a one-class classifier is also related to its performance on the target and outlier

class. In general, the larger the volume of a classifier the better the performance on the target class. Therefore, selection of models based on the minimisation of r in the criterion (6.1) does not allow for good generalisation of the classifier on the target class. In general, a large volume of a one-class classifier indicates good generalisation on the target class, however bad rejection of outlier objects. A small volume of a classifier, however, indicates bad generalisation on the target class and a good rejection of outlier objects.

Since we want to confer both the error on the target and outlier class we consider both errors, and we propose to minimise the ratio of two volumes: the volume of the largest empty N-sphere $V_{\mathcal{S}(\mathbf{a},r)}$, inside a classifier divided by the volume of the classifier V_h itself:

$$\mathcal{V} = \frac{V_{\mathcal{S}(\mathbf{a},r)}}{V_h} = \frac{2\pi^{N/2}}{\Gamma(N/2+1)} \frac{r^N}{V_h} \tag{6.2}$$

Based on the proposed criterion we select classifiers with a large volume, therefore with a good generalisation on a target class, and a small volume of the largest empty N-sphere, therefore penalising the region of the classifier which can be consider superfluous. We denote the above ratio the $\mathcal{V}$-statistic. The $\mathcal{V}$-statistic has values in the range $[0, 1]$. Such criterion can be used as a measure of how well a classifier fits the target class distribution. Note that when the $\mathcal{V}$-statistic increases, we expect an increase in the outlier acceptance rate ε_o or an increase in target rejection rate ε_t. For a fixed value of the threshold θ as $V_{\mathcal{S}(\mathbf{a},r)}$ and V_h depends on γ also the $\mathcal{V}$-statistic depend on γ. For the spherical one-class classifiers in figure 6.1, the $\mathcal{V}$-statistic reduces to the ratio of two radii $(\frac{r}{R})^N$. The problem of selecting a complexity parameter in terms of the volume of a one-class classifier can be considered as the specialisation/generalisation trade-off [Bishop, 1995] on the target class.

In relation to the method of uniform outlier generation, proposed in [Tax and Duin, 2001], we do not aim at the minimisation of the volume of the entire one-class classifier, but only this region of the classifier which is empty, therefore not supported by the training objects, and can be considered superfluous. Such an empty region can be the result of adopting an inappropriate, too simple classification model. On the other hand, too small volume of the classifier can be the result of selecting a too complex classifier.

By deciding to minimise the $\mathcal{V}$-statistic we assume that the data is well scaled and that the empty region inside the classifier is well approximated by an N-sphere. In situations where an empty region inside classifier has e.g. an ellipsoidal shape the criterion can be modified by considering several empty and non-intersecting N-spheres with maximum volume; see figure 6.2.

A set of nonintersecting N-spheres can be computed by posing an additional constraint in (6.1) on the next added N-sphere $\mathcal{S}(\mathbf{a}_q, r_q)$:

$$\mathcal{I}\{\|\mathbf{a}_q - \mathbf{a}_{q-1}\| \geq r_q + r_{q-1}\} \wedge \ldots \wedge \mathcal{I}\{\|\mathbf{a}_q - \mathbf{a}_1\| \geq r_q + r_1\}$$

Therefore, the criterion (6.2) can be defined as:

$$\mathcal{V}(q) = \frac{\sum_{i=1}^{q} V_{\mathcal{S}(\mathbf{a}_i,r_i)}}{V_h} = \frac{2\pi^{N/2}}{\Gamma(N/2+1)} \frac{\sum_{i=1}^{q} r_i^N}{V_h} \tag{6.3}$$

The number of N-spheres q can be selected by comparing volumes of currently and previously computed empty and non-intersecting N-spheres or based on, e.g. the maximum nearest neighbour distance of objects in a training set. Therefore, q is increased if the volume of the $(q+1)$th largest empty and nonintersecting, with previous q N-spheres, an N-sphere is significantly larger or if the radius of the $(q+1)$th N-sphere is larger than the maximum nearest neighbour distance.

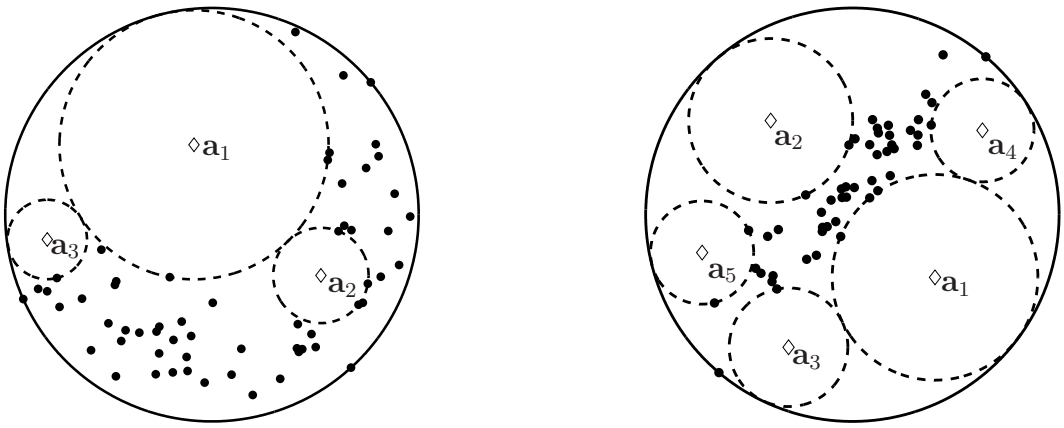


Figure 6.2: Several empty and nonintersecting N-spheres that can be found inside the classifier determined as the smallest N-sphere enclosing a training set.

6.1 The estimation of $\mathcal{V}$-statistic

To estimate the $\mathcal{V}$-statistic we have to know the volume of the classifier V_h. Therefore, we make a distinction into two groups of classifiers. The first group is based on a union of k open N-spheres $h \equiv \bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$, where the classification is formulated as a logic union: $\mathcal{I}(h(\mathbf{x}|X_t, \theta)) = \bigvee_{i=1}^{k} \mathcal{I}\{||\mathbf{A}_i - \mathbf{x}|| < R_i\}$, where $||\cdot||$ is the Euclidean norm. This holds for some types of one-class classifiers, in particular: the single Gaussian with equal elements on a diagonal covariance matrix, k-means, k-centres, self-organising maps (`SOM`) and the support vector based method: `SVDD` with a linear kernel. Consequently, these one-class classifiers can be described either by a single N-sphere or by the union of several N-spheres. We show how to compute the volume of such classifiers as well as the determination of the largest empty N-sphere inside a classifier. The second group consists of non-spherical one-class classifiers, e.g. `SVDD`, *oc*-`SVM` with a nonlinear kernel, Parzen, mixture of Gaussians or an auto-encoder. For this group of classifiers we present approximate solutions.

Actually, there is also a third group of one-class classifiers, which instead of bounding a finite volume of a space, describe an infinite subspace, e.g. the `PCA` one-class classifier [Tax, 2001]. These methods are not considered here.

In the rest of this section we describe how to compute the volume of a classifier that consists of several N-spheres and how to find the largest empty N-sphere inside such classifier. In addition, in subsection 6.1.2 we provide explanation why the proposed algorithm to find the largest, empty N-sphere does not work for `SVDD` and *oc*-`SVM` with a nonlinear kernel. We finalise the section by presenting an algorithm to find the approximately largest empty N-sphere in the one-class classifiers that belong to the second group.

6.1.1 Estimation of the volume of one-class classifiers

In general, algorithms that approximate the volume of convex or non-convex bodies are not strongly polynomial in N. The best known algorithms for convex bodies are of the order $\mathcal{O}(N^5)$; see for example [Lovász et al., 1997] for a survey. In [Tax and Duin, 2001] an algorithm that can be used to approximate the volume of non-convex or convex one-class classifiers has been proposed. The volume is estimated from the fraction of uniformly distributed objects that are classified as targets. Objects are generated in the smallest N-sphere, that enclose all

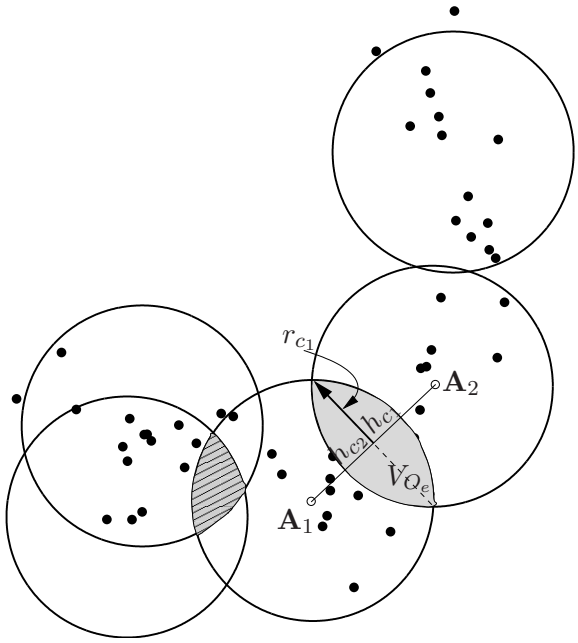


Figure 6.3: 5-means classifier with $\varepsilon_t^{tr} = 0.05$. The two spherical caps are painted gray and the region that belongs to more than two spheres is painted gray and crossed.

data; see Algorithm 5.1 for details. However, such an algorithm requires a large number of objects to accurately approximate the volume of a classifier V_h. Therefore the method is only applicable for low dimensional problems, up to $N = 15$ or 20, for $100,000$ generated outlier objects, depending on the distribution of target data. However, if we assume that a one-class classifier is based on a union of several open N-spheres $h \equiv \bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$, the computation of its volume simplifies significantly.

If N-spheres that constitute the classifier do not intersect, the volume of the classifier is simply equal to the sum of volumes of the individual N-spheres:

$$V_h = \sum_{i=1}^{k} V_{S(\mathbf{A}_i,R_i)} \quad \Leftrightarrow \quad \forall_{\substack{i\neq j \\ i,j=1,\ldots,k}} \quad V_{S(\mathbf{A}_i,R_i)} \cap V_{S(\mathbf{A}_j,R_j)} = \emptyset \tag{6.4}$$

When the N-spheres that constitute the classifier only intersect pairwise, which means there is no region that belongs to more than two N-spheres, the volume of the classifier can be computed as the sum of the volumes of individual N-spheres minus the sum of volumes of their pairwise overlaps $\sum_e V_{O_e}$:

$$\begin{aligned} &V_h = \sum_{i=1}^{k} V_{S(\mathbf{A}_i,R_i)} - \sum_{e} V_{O_e} \Leftrightarrow \\ &\forall_{\substack{i\neq j\neq q \\ i,j,q=1,\ldots,k}} \{V_{S(\mathbf{A}_i,R_i)} \cap V_{S(\mathbf{A}_j,R_j)} \neq \emptyset\} \wedge \{V_{S(\mathbf{A}_i,R_i)} \cap V_{S(\mathbf{A}_j,R_j)} \cap V_{S(\mathbf{A}_q,R_q)} = \emptyset\} \end{aligned} \tag{6.5}$$

The volume of a single overlap V_{O_e} between two N-spheres equals the sum of volumes of two spherical caps. The spherical cap [Harris and Stocker, 1998] is a part of an N-sphere defined by its height $h_c \in [0, 2R]$ and radius $r_c \in [0, 2R]$; see figure 6.3 where the overlap, the gray region, is created from two spherical caps between spheres $\mathcal{S}(\mathbf{A}_1, R_1)$ and $\mathcal{S}(\mathbf{A}_2, R_2)$.

Note that these two spherical caps have the same radius r_c but different heights h_c. We have derived (see Appendix B.1 for a derivation) the volume of a N dimensional spherical cap as an integral of $N-1$ dimensional spheres over the height h_c of the cap:

$$V_{cap}(R, h_c) = \frac{\pi^{(N-1)/2} R^{N-1}}{\Gamma((N-1)/2+1)} \int_0^{\beta_{max}} \sin^{N-1}(\beta) d\beta \tag{6.6}$$

where

$$\beta_{max} = \arcsin(\sqrt{(2R - h_c)(h_c/R^2)})$$

Therefore the volume of a single overlap $\mathcal{V}_O$ between two N-spheres can be computed as the sum of two spherical caps:

$$V_O = V_{cap}(R_1, h_{c_1}) + V_{cap}(R_2, h_{c_2})$$

When a classifier consists of N-spheres intersecting in an arbitrary way the computation of the volume of the classifier becomes more difficult than in the case when N-spheres intersect only pairwise. However, if we compute the volume of such classifier using equation (6.5), the volume of a region inside h that belongs to more than two N-spheres is subtracted several times. Therefore, the volume of the classifier is underestimated. The example of such region is shown in figure 6.3 as the gray-crossed region.

Note however that the volume V_h of a classifier consisting of arbitrarily intersecting N-spheres can be bounded between the two volumes computed in equations (6.4) and (6.5), namely:

$$\sum_{i=1}^{k} V_{S(\mathbf{A}_i, R_i)} - \sum_e V_{O_e} \;\leq\; V_h \;\leq\; \sum_{i=1}^{k} V_{S(\mathbf{A}_i, R_i)} \tag{6.7}$$

Based on these bounds, we approximate the volume of a classifier by:

$$V_h \approx \sum_{i=1}^{k} V_{S(\mathbf{A}_i, R_i)} - \frac{1}{2} \sum_e V_{O_e} \tag{6.8}$$

The maximum error of estimating V_h in this way is equal to half the volume of all pairwise intersections, i.e. $\frac{1}{2}\sum_e V_{O_e}$. This approximation can be improved by checking, whether a region of intersection belongs to more than two N-spheres. We proposed a simple check based on quadratic programming (see Appendix B.2 for derivation). For k N-spheres with centres $\mathbf{A}_i$ and radii R_i, $i = 1, 2, \ldots, k$ the check is expressed as:

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{A}_1 + \mathbf{A}_1^T\mathbf{A}_1, \\ \text{st.} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{A}_i + \mathbf{A}_i^T\mathbf{A}_i - R_i^2 < 0, \qquad i = 2, \ldots, k. \end{aligned} \tag{6.9}$$

if $\mathbf{y}$ exists and $\|\mathbf{y} - A_1\| < R_1$ then there exist a region where these k N-spheres overlap.

The approximation of the volume of a classifier consisting of arbitrary overlapping N-spheres can be written as:

$$\begin{aligned} &\exists_{\substack{i\neq j\neq q \\ i,j,q=1,\ldots,k}} \{V_{S(\mathbf{A}_i, R_i)} \cap V_{S(\mathbf{A}_j, R_j)} \cap V_{S(\mathbf{A}_q, R_q)} \neq \emptyset\} \\ &\Rightarrow V_h \approx \sum_{i=1}^{k} V_{S(\mathbf{A}_i, R_i)} - \sum_{e1} V_{O_{e1}} - \frac{1}{2} \sum_{e2} V_{O_{e2}} \end{aligned} \tag{6.10}$$

where $\sum_{e1} V_{O_{e1}}$ is the volume of overlap of N-spheres that overlap only pairwise and $\frac{1}{2}\sum_{e2} V_{O_{e2}}$ is the approximated volume of regions where more than two N-spheres overlap. When estimating the volume of the classifier, in this way the largest error we can make is for regions that belongs to all N-spheres. The largest positive error is less than half the volume of pairwise overlaps when all N-spheres have the same radius and their centres are almost the same. In such a case, the volume V_h is underestimated by $\frac{1}{2}\sum_{e2} V_{O_{e2}}$, which can not be larger than the half of the volume of entire classifier. The largest negative error in estimating V_h using equation (6.10) is when all N-spheres have infinitely small common region. The volume of a classifier is than overestimated by $\frac{1}{2}\sum_{e2} V_{O_{e2}}$, which also can not be larger than the half of the volume of a classifier. However, the volume $\sum_{e2} V_{O_{e2}}$ is usually several orders smaller than the volume of a classifier, especially in high dimensional spaces.

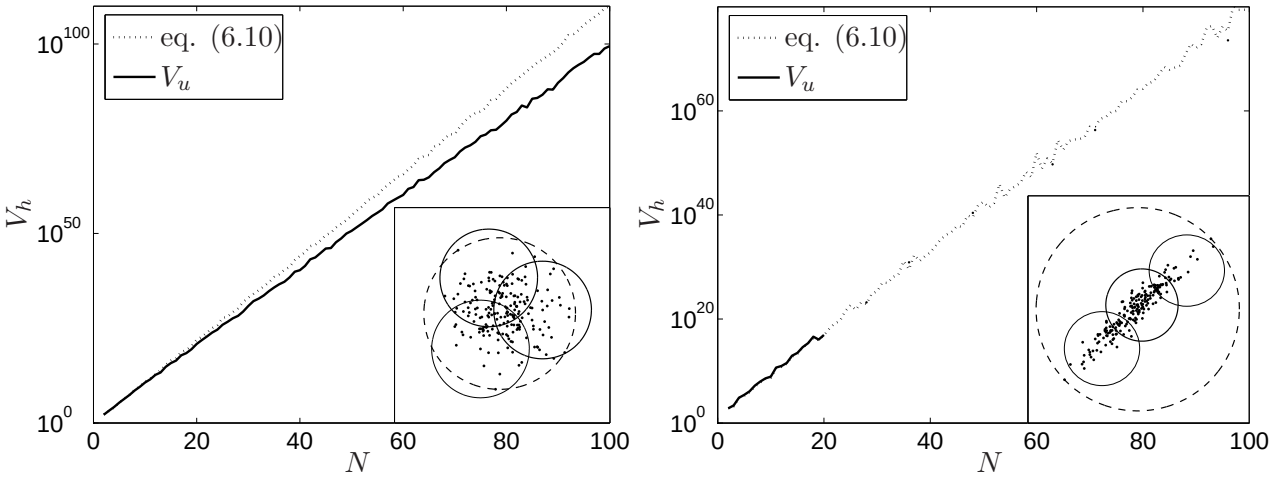


Figure 6.4: The estimate of the volume V_h using uniformly distributed objects (V_u) compared with the volume calculated using equation (6.10). The volume is shown in the logarithmic scale. A 3-centres classifier is trained on uncorrelated (a) and correlated (b) normal distributed data with the number of dimensions varied from $N = 2$ to 100. The smallest N-sphere that encloses the training data is denoted by dashed line.

Now, we can compare the volume V_h as estimated by the proposed equation (6.10) with the volume estimated by the Algorithm 5.1 using uniformly distributed objects [Tax and Duin, 2001] denoted as V_u. In figure 6.4, we plotted the estimated volumes, using the proposed method and Algorithm 5.1, for the 3-centres classifier trained on two different datasets. In the left subplot the classifier is computed on normal, uncorrelated data $\mathcal{N}(\mathbf{0}, 5I)$ and in the right subplot on data with a strong correlation between the two first features, the all other features are distributed as $\mathcal{N}(0, 1)$. Both datasets sizes equal 200. The dimensionality N of the data was varied between 2 to 100. We use $100,000$ uniformly distributed objects in Algorithm 5.1 for all values of N to estimate the volume of the classifier. This number was also used by the authors in [Tax and Duin, 2001] as the maximum number of uniformly generated objects.

From the figure 6.4(a) it can be seen that the estimated volumes differ when N increases. For increasing dimensionality the probability that a uniformly distributed object, from a finite cardinality set, is inside the classifier decreases. For a finite number of uniformly distributed

objects generated in the N-sphere computed by a linear-SVDD (dashed line in figure 6.4) the probability that a uniformly distributed object is inside the classifier is inversely proportional to the volume of that sphere. Therefore, the difference between estimated volumes increases systematically with N. For large N the volume mass of an N-sphere is distributed on its surface therefore uniformly distributed objects are also on the surface of an N-sphere.

However, because the classifier is almost spherical, some of the generated objects are always encountered inside the 3-centres classifier as parts of the smallest N-sphere that enclosed data is inside the classifier. The N-sphere where we generate objects is always inside the classifier; see three moon shaped regions in figure 6.5. For the second dataset for $N > 20$ uniformly distributed objects, from the finite set, are not encountered inside the classifier resulting in zero estimate for V_u. The volume of the classifier compared to the volume of the N-sphere determined by a smallest N-sphere enclosing data is significantly larger.

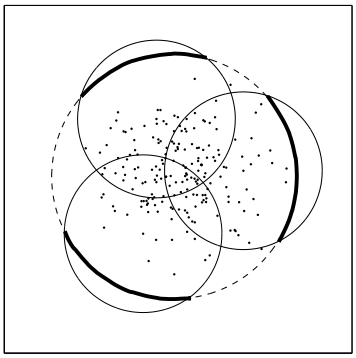


Figure 6.5: 3-centre classifier. The parts that are inside the smallest enclosing sphere are marked by thick lines.

From this experiment we conclude that the method based on the uniform objects generation can be used up to 15-20 dimensions, with $100,000$ uniformly generated objects. The similar conclusion was given in [Tax and Duin, 2001]. We provide relations between the number of objects, data dimensionality and error on the estimation of the volume of a classifier in section 6.1.4.

From here, we use equation (6.10) to estimate the volume of classifiers consisting of an arbitrary number of N-spheres. For non-spherical one-class classifiers, e.g. Parzen or SVDD with the nonlinear kernel we use Algorithm 5.1 to estimate their volumes in low dimensional, up to 20, recognition problems.

6.1.2 Estimation of the largest empty N-sphere inside spherical one-class classifiers

In the previous subsection we have shown how to approximate in high dimensional spaces the volume of a classifier consisting of the union of N-spheres. We have also discussed how to estimate the volume of an arbitrary classifier using uniformly generated objects in a low number of dimensions. These methods compute the denominator in the $\mathcal{V}$-statistic; see equation (6.2). Now we focus on the problem of how to find the largest empty N-sphere inside a classifier. We start from the simplest case when a classifier is based on a single N-sphere.

Estimation of the largest empty N-sphere inside a classifier consisted of a single N-sphere

Given a target set X_t in $\mathbb{R}^N$, a one-class classifier is modelled now by an N-sphere $\mathcal{S}(\mathbf{A}, R)$; see figure 6.6(a). $\mathcal{S}(\mathbf{A}, R)$ can be determined by e.g. a single Gaussian with equal elements on a diagonal covariance matrix, 1-centre algorithm, SOM or linear-SVDD. Our task in this subsection

is to find the largest N-sphere $\mathcal{S}(\mathbf{a}, r)$ such that $\mathcal{S}(\mathbf{a}, r) \subset \mathcal{S}(\mathbf{A}, R)$ and $\forall_{\mathbf{x}_i \in X_t} \|\mathbf{a} - \mathbf{x}_i\| \geq r$; see figure 6.6(f).

Since $\mathcal{S}(\mathbf{A}, R)$ is known, then the problem of finding the largest, empty $\mathcal{S}(\mathbf{a}, r) \subset \mathcal{S}(\mathbf{A}, R)$ can be captured by formulating the following optimisation problem:

$$\begin{aligned} \max \quad & r && \text{(6.11a)} \\ \text{s.t.} \quad & \|\mathbf{a} - \mathbf{A}\| + r \leq R && \text{(6.11b)} \\ & \|\mathbf{a} - \mathbf{x}_i\| \geq r, \qquad \forall\ \mathbf{x}_i \in X_t && \text{(6.11c)} \end{aligned}$$

The objective function (6.11a) is linear in the variable r and the constraint (6.11b) is a second order cone constraint. Such a problem can be solved by second order cone programming [Lobo et al., 1998]. The constraint (6.11c), however, is non-convex. To see it, let us write $\mathbf{t}_i = \mathbf{a} - \mathbf{x}_i$. Then, by squaring both sides and subtracting r^2 we get $||\mathbf{t}_i||^2 - r^2 \geq 0$. We replace $||\mathbf{t}_i||^2 - r^2$ by $\tilde{\mathbf{t}}_i^T J \tilde{\mathbf{t}}_i$, where $J = \begin{bmatrix} I & 0 \\ 0 & -1 \end{bmatrix}$, I is an $n \times n$ identity matrix and $\tilde{\mathbf{t}}_i = \begin{bmatrix} \mathbf{t}_i \\ r \end{bmatrix}$. The condition (6.11c) becomes then

$$\tilde{\mathbf{t}}_i^T J \tilde{\mathbf{t}}_i \geq 0, \quad i = 1, 2, \ldots, n$$

which is a non-convex quadratic constraint, as J is not positive semidefinite.

Since the problem is non-convex, it can not be solved using quadratic programming. For the problem of determining the smallest N-sphere $\mathcal{S}(\mathbf{A}, R)$ that contains all data no strongly polynomial algorithm is known [Gritzmann and Klee, 1993]. However, the problem of determining the largest empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ inside a larger N-sphere $\mathcal{S}(\mathbf{A}, R)$ as we can see is even harder. Since the problem (6.11) is locally convex it has several local maxima. Each local maximum is an N-sphere determined by $N+1$ objects in an N dimensional space. Our proposition is to search for the global optimum by computing and comparing the set of the most likely local maxima.

Note first that the centre of the largest empty N-sphere is located in the vicinity of either a local or a global minimum of a density function $f(\mathbf{z})$, where $f(\mathbf{z})$ is the density of the training set and objects on the surface of $\mathcal{S}(\mathbf{A}, R)$; figure 6.6(b). Therefore, we minimise $f(\mathbf{z})$ starting from an object $\mathbf{x}_i \in X_t$ and determine an approximate centre $\mathbf{z}_i^{(1)}$ for each $\mathbf{x}_i$. Next, we compute the equidistant object $\mathbf{z}_i^{(2)}$ to $N+1$ nearest neighbours of $\mathbf{z}_i^{(1)}$ from $\tilde{X} = \{X_t \cup \mathbf{p}(\mathbf{z}_i^{(1)})\}$. Where $\mathbf{p}(\mathbf{z}_i^{(1)})$ is the projection of $\mathbf{z}_i^{(1)}$ onto the surface of $\mathcal{S}(\mathbf{A}, R)$. Finally, we select that centre, which has the largest nearest neighbour distance to objects in $\tilde{X}$. This determines the largest empty N-sphere $\mathcal{S}(\mathbf{a}, r)$. The proposed algorithm is described below and it is schematically presented in Algorithm 6.1 and figure 6.6 for a single optimisation, starting from an initial object $\mathbf{x}_i \in X_t$.

Notice first that $N+1$ non-collinear points $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\} \subset X_t$ uniquely define an N-sphere $\mathcal{S}(\mathbf{a}, r)$ in $\mathbb{R}^N$. In a 1-dimensional space, the centre a of such a sphere is equidistant from two given points x_1 and x_2, i.e. $\|x_1 - a\|^2 = \|x_2 - a\|^2$. After easy computations, one finds that $a = \frac{1}{2}((x_1 - x_2)^2)^{-1}(x_1 - x_2)(x_1^2 - x_2^2)$. In the N-dimensional space, the centre $\mathbf{a}$ is determined as (see Appendix B.3 for a derivation):

$$\mathbf{a} = \frac{1}{2}(X_- X_-^T)^{-1} X_- (\mathbf{1}^T X_-^T X_+)^T. \tag{6.12}$$

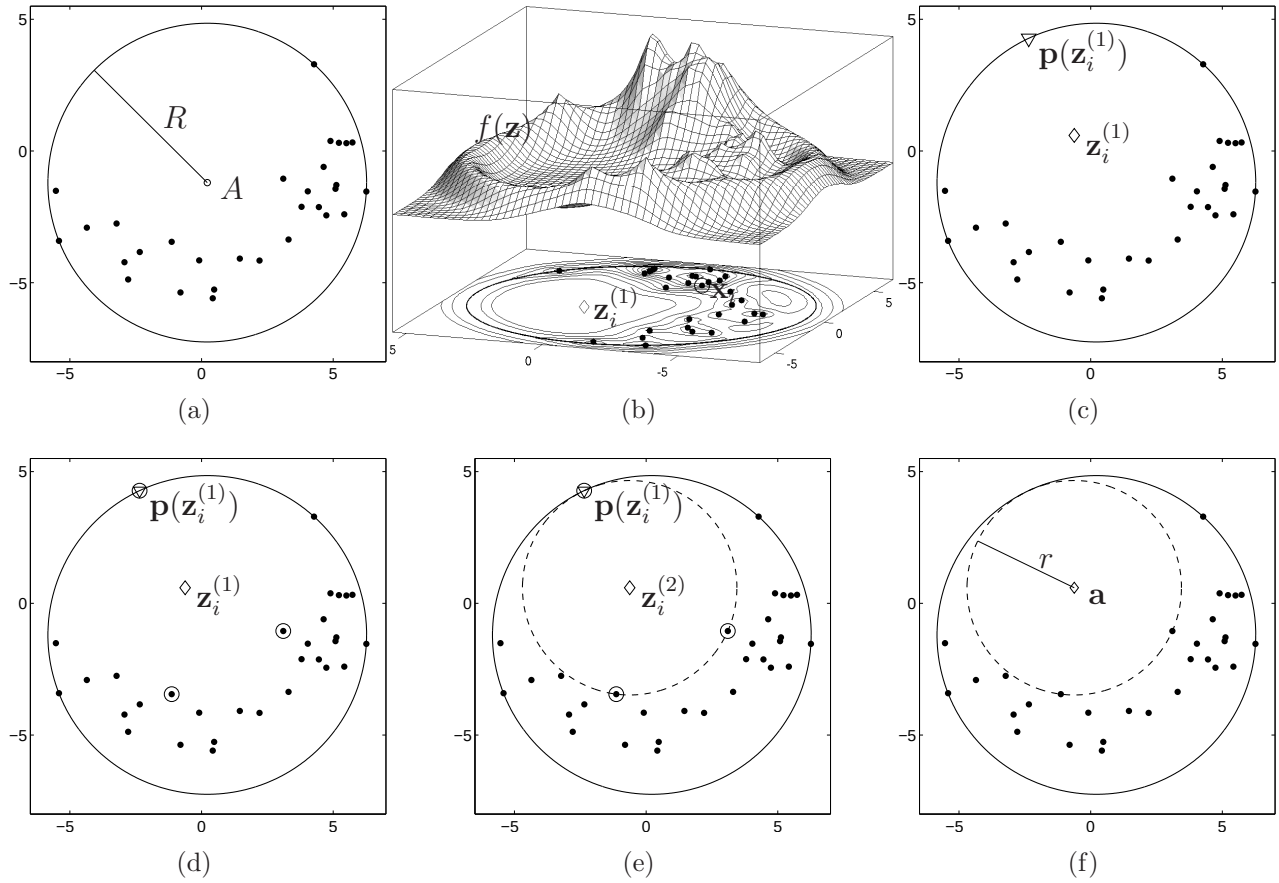


Figure 6.6: Graphical illustrations of Algorithm 6.1. $\mathbf{z}_i^{(\cdot)}$ is a current estimation of $\mathbf{a}$ and $\mathbf{p}(\mathbf{z}_i^{(\cdot)})$ is its projection onto the surface of $\mathcal{S}(\mathbf{A}, R)$. $\mathbf{x}_i$ in figure 6.6(b) is an initial vector for a minimisation of a function $f(\mathbf{z})$.

where X_- is a $N \times \binom{N+1}{2}$ matrix of elements $\mathbf{x}_i - \mathbf{x}_j$, X_+ is a $N \times \binom{N+1}{2}$ matrix of elements $\mathbf{x}_i + \mathbf{x}_j$ and $\mathbf{1}$ is a $\binom{N+1}{2} \times 1$ vector of ones.

Having found $\mathbf{a}$, the radius r of $\mathcal{S}(\mathbf{a}, r)$ is derived by computing the distance between $\mathbf{a}$ and any of the given points $\mathbf{x} \in \{\mathbf{x}_1, \ldots, \mathbf{x}_{N+1}\}$:

$$r = ||\mathbf{a} - \mathbf{x}|| \tag{6.13}$$

In the considered problem of finding the largest, empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ inside an N-sphere $\mathcal{S}(\mathbf{A}, R)$, one of the points that define $\mathcal{S}(\mathbf{a}, r)$ may happen to be an unknown intersection of the two N-spheres $\mathcal{S}(\mathbf{A}, R)$ and $\mathcal{S}(\mathbf{a}, r)$; see object $\mathbf{p}(\mathbf{z}_i^{(1)})$ in figure 6.6(e) for an illustration. If $\mathbf{a}$ was known, then the intersection point $\mathbf{p}(\mathbf{a})$ can be easily computed as the projection of $\mathbf{a}$ onto $\mathcal{S}(\mathbf{A}, R)$:

$$\mathbf{p}(\mathbf{a}) = R \frac{\mathbf{a} - \mathbf{A}}{\|\mathbf{a} - \mathbf{A}\|} + \mathbf{A}. \tag{6.14}$$

Since $\mathbf{p}(\mathbf{a})$ is unknown we can not simply determine all combination of $N+1$ neighbouring objects using equation (6.12) and find the centre with the largest r in equation (6.13). However, it should be noticed that centres of empty N-spheres are in a region of low density of training objects and the surface of $\mathcal{S}(\mathbf{A}, R)$. Therefore, we can first find approximations of the centre

of the empty N-sphere by minimising the the following density function:

$$f(\mathbf{z}_i) = \sum_{j=1}^{n} \exp\left(-\frac{\|\mathbf{z}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) + \exp\left(-\frac{(R - \|\mathbf{z}_i - \mathbf{A}\|)^2}{2\sigma^2}\right), \quad \mathbf{x}_j \in X_t. \tag{6.15}$$

The first part of the above equation is simply the Parzen density estimation [Parzen, 1962] with a Gaussian kernel. The second part is proportional to the distance of $\mathbf{z}_i$ to the surface of the N-sphere $\mathcal{S}(\mathbf{A}, R)$: $R - \|\mathbf{z}_i - \mathbf{A}\|$. Vectors $\mathbf{z}_i$ for which $f(\mathbf{z}_i)$ has small values are far from training objects X_t and from the surface of $\mathcal{S}(\mathbf{A}, R)$. However, the result of minimisation of the function (6.15) depends on initial vectors. In our algorithm we perform n optimisations using as initialisations $\mathbf{x}_i \in X_t$ individually. Moreover, the result of the minimisation depends on the value of σ. For large σ, $\mathbf{z}_i$ is determined by a large number of objects from X_t, for small σ only by nearest neighbours. It is hard to decide on the correct value of σ, therefore, first we perform the minimisation using large σ, e.g. equal to the largest nearest neighbour distance in X_t, and then decrease σ until the value of function $f(\mathbf{z}_i) \approx 0$, giving the searched $\mathbf{z}_i$. The first and second derivative of the function (6.15) are easily computable, therefore it can be minimised using most optimisation toolboxes. We minimise (6.15) using the MATLAB implementation of large scale optimisation of the inferior-reflection Newton method [Coleman and Li, 1996] with the constrain $\|\mathbf{z}_i - \mathbf{A}\| < R$, which constrained the set of solutions to those in $\mathcal{S}(\mathbf{A}, R)$:

$$\min_{\mathbf{z}} \quad f(\mathbf{z})_{\mathbf{x}_i}, \tag{6.16a}$$

$$\text{s.t.} \quad \mathbf{z}^T\mathbf{z} - 2\mathbf{z}^T\mathbf{A} + \mathbf{A}^T\mathbf{A} - R^2 < 0, \quad \forall \mathbf{z}_i \in \mathbb{R}^N. \tag{6.16b}$$

Now the full algorithm can be described as follows.

Assume a set of objects $X_t = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ in $\mathbb{R}^N$, $n \geq N+1$. Let $\mathbf{A}$ and R be the known parameters of an N-sphere $\mathcal{S}(\mathbf{A}, R)$ encapsulating X_t completely.

1. Starting from $\mathbf{x}_i \in X_t$ find an object $\{\mathbf{z}_i^{(1)} : \|\mathbf{z}_i^{(1)} - \mathbf{A}\| < R\}$ for which a density function $f(\mathbf{z})$, equation (6.15), has a minimum; $\mathbf{z}_i^{(1)} = \arg\min_{\mathbf{z}} f(\mathbf{z})_{\mathbf{x}_i}$ (using (6.16)).
2. Project $\mathbf{z}_i^{(1)}$ on the surface of the N-sphere $\mathcal{S}(\mathbf{A}, R)$ as specified by (6.14). Add the projected point $\mathbf{p}(\mathbf{z}_i^{(1)})$ to the set X_t, obtaining $\tilde{X} = \{X_t \cup \mathbf{p}(\mathbf{z}_i^{(1)})\}$.
3. Find the set $\tilde{X}_{N+1}(\mathbf{z}_i^{(1)})$ consisting of $N+1$ nearest neighbours of $\mathbf{z}_i^{(1)}$ in $\tilde{X}$. Use formula (6.12) to define the N-sphere $\mathcal{S}(\mathbf{z}_i^{(2)}, \rho_i^{(2)})$ determined by the $N+1$ objects in $\tilde{X}_{N+1}(\mathbf{z}_i^{(1)})$.
4. Repeat step 3 until the centre of the N-sphere does not change yielding N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$
5. Check whether the N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$ is inside the N-sphere $\mathcal{S}(\mathbf{A}, R)$, i.e. if the condition holds
$$\rho_i + \|\mathbf{z}_i - \mathbf{A}\| \leq R \tag{6.17}$$

Algorithm 6.1: Find an empty N-sphere inside a given N-sphere $\mathcal{S}(\mathbf{A}, R)$ starting from an initial object $\mathbf{x}_i$.

To determine the largest empty N-sphere inside a one-class classifier, modelled by a single

N-sphere $\mathcal{S}(\mathbf{A}, R)$; figure 6.6(a), we follow the procedure described in Algorithm 6.1. First, an object $\mathbf{z}_i^{(1)}$ is determined by the minimisation of $f(\mathbf{z})$ in (6.16), as a current approximation of the centre $\mathbf{a}$ of $\mathcal{S}(\mathbf{a}, r)$; figure 6.6(b). Since one of the $N+1$ points defining the largest, empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ may happen to be an intersection of two N-spheres, $\mathcal{S}(\mathbf{A}, R)$ and $\mathcal{S}(\mathbf{a}, r)$, the object $\mathbf{z}_i^{(1)}$ is projected onto $\mathcal{S}(\mathbf{A}, R)$ using equation (6.14); figure 6.6(c). We use the projection $\mathbf{p}(\mathbf{z}_i^{(1)})$ as the current estimation of the possible intersection point of the two N-spheres. We add the projected point $\mathbf{p}(\mathbf{z}_i^{(1)})$ to the training set X_t, $\tilde{X} = \{X_t \cup \mathbf{p}(\mathbf{z}_i^{(1)})\}$. In step 3 of Algorithm 6.1 we determine the set $\tilde{X}_{N+1}$ of $N+1$ nearest neighbours of $\mathbf{z}_i^{(1)}$ in $\tilde{X}$, figure 6.6(d), and compute a centre $\mathbf{z}_i^{(2)}$ and the radius $\rho_i^{(2)}$ of the N-sphere defined by the objects in $\tilde{X}_{N+1}$ using equation (6.12), figure 6.6(e). We repeat the previous step until the centre of the N-sphere does not change; figure 6.6(e). In step 5 we check if the N-sphere determined by $N+1$ nearest neighbours is inside $\mathcal{S}(\mathbf{A}, R)$ [1]. Since $\mathbf{z}_i$ can converge to a local minimum Algorithm 6.1 is performed for all n initialisations $\mathbf{x}_i \in X_t$. From all N-spheres that satisfy equation (6.17) the one with the largest ρ_i is the sought solution $\mathcal{S}(\mathbf{a}, r)$; figure 6.6(f). If $\mathcal{S}(\mathbf{a}, r)$ can not be found using Algorithm 6.1, e.g. because there are less than $N+1$ objects in $\mathcal{S}(\mathbf{A}, R)$ or all N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i)$ are outside $\mathcal{S}(\mathbf{A}, R)$ we determine an N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$ based on the single nearest neighbour distance between $\mathbf{z}_i$ and objects in $\tilde{X}$. $\mathcal{S}(\mathbf{a}, r)$ is the sphere with maximum radius among all $\mathcal{S}(\mathbf{z}_i, \rho_i)$.

Estimation of the largest empty *N*-sphere inside one-class classifier consisting of k *N*-spheres

In this section, we extend Algorithm 6.1 to one-class classifiers constructed from a union of several N-spheres $h \equiv \bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$ e.g. k-means, k-centres `SOM` and nearest neighbour. Algorithm 6.1 can be applied to each of the k N-spheres of a classifier. However, there are situations where the largest empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ is not entirely inside one of the k N-spheres, that is a part of the classifier, but it is still inside the classifier; see figure 6.7. We can see that in this situation more than one $\mathcal{S}(\mathbf{A}_i, R_i)$ should be considered. Therefore, we propose to replace the inequality (6.17) in Algorithm 6.1 by the solution to the quadratic programming problem defined in (6.18).

[1] Most of the of N-spheres determined by Algorithm 6.1 are in $\mathcal{S}(\mathbf{A}, R)$, although there are some degenerative cases where $\mathcal{S}(\mathbf{z}_i, \rho_i)$ can be outside $\mathcal{S}(\mathbf{A}, R)$.

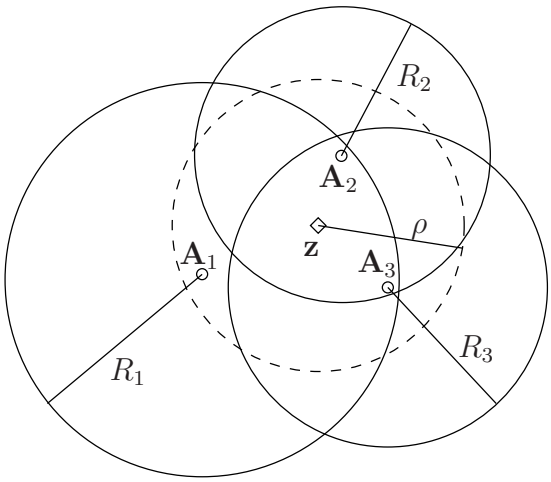


Figure 6.7: Largest empty N-sphere (dashed line) inside a one-class classifier constructed from three spheres $\bigcup_{i=1}^{3} \mathcal{S}(\mathbf{A}_i, R_i)$ (continues line), data are omitted for clarity.

We would like to find the vector $\mathbf{y}$ that is inside $\mathcal{S}(\mathbf{z}, \rho)$ and that is outside the classifier defined by the union of k N-spheres $\bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$.

$$\begin{aligned} \|\mathbf{y} - \mathbf{z}\|^2 &\leq \rho^2 \\ \|\mathbf{y} - \mathbf{A}_i\|^2 &\geq R_i^2, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{6.18}$$

This set of inequalities can be replaced by the following quadratic minimisation problem (see Appendix B.4 for a derivation):

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{z} \\ \text{s.t.} \quad & 2\mathbf{y}^T(\mathbf{A}_i - \mathbf{z}) \leq \rho^2 + \mathbf{A}_i^T\mathbf{A}_i - R_i^2 - \mathbf{z}^T\mathbf{z}, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{6.19}$$

If the solution $\mathbf{y}$ exists and it satisfies the first inequality in (6.18) then the N-sphere $\mathcal{S}(\mathbf{z}, \rho)$ is outside the union of N-spheres $\bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$.

Now we are able to compute the $\mathcal{V}$-statistic for one-class classifiers consisting of N-spheres. Consider the $2D$ toy examples of novelty detection problems shown in figure 6.9. Three spherical one-class classifiers are trained k-centres, k-means and `SOM`. On all problems, we set the target rejection rate to $\varepsilon_t^{tr} = 0.01$, i.e. the threshold θ is set in such a way that $\varepsilon_t^{tr} = 0.01$. The question rises: How complex these three classifiers should be to generalise on the target class and reject large amount of outlier objects.

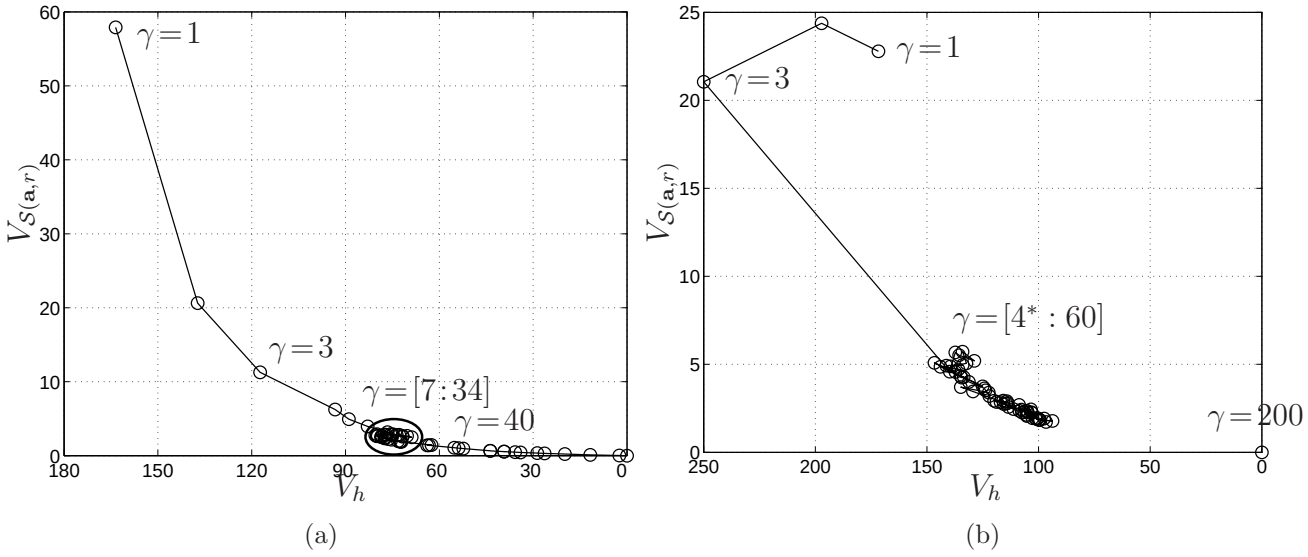


Figure 6.8: Volume of a k-centres (a) and k-means (b) classifiers against volume of the largest empty sphere for different value of γ. The complexity γ increases from left to right.

For the given datasets we estimate the $\mathcal{V}$-statistic specified by equation (6.2) using Algorithm 6.1 to find $\mathcal{S}(\mathbf{a}, r)$ and equation (6.10) to compute V_h. Since the volume estimation of

a classifier and the largest, empty N-sphere might depend on outliers, we average $\mathcal{V}$ over 10 random draws of 90% of the training data. The range of γ values are selected between 1 and 60. The value of $\mathcal{V}$ against the complexity of classifiers γ is shown in figure 6.9, together with some 2 dimensional plots of the three classifiers and $\mathcal{S}(\mathbf{a}, r)$. It can be seen that when we increase the complexity of these three classifiers the $\mathcal{V}$-statistic first decreases fast and after a certain value of γ almost stabilises. The reason for this behaviour can be seen in figure 6.8 where, as an example, the volumes of the k-centre and k-means classifiers are plotted against the volume of the largest empty N-sphere. The $\mathcal{V}$-statistic in this plot is the tangent of the curve.

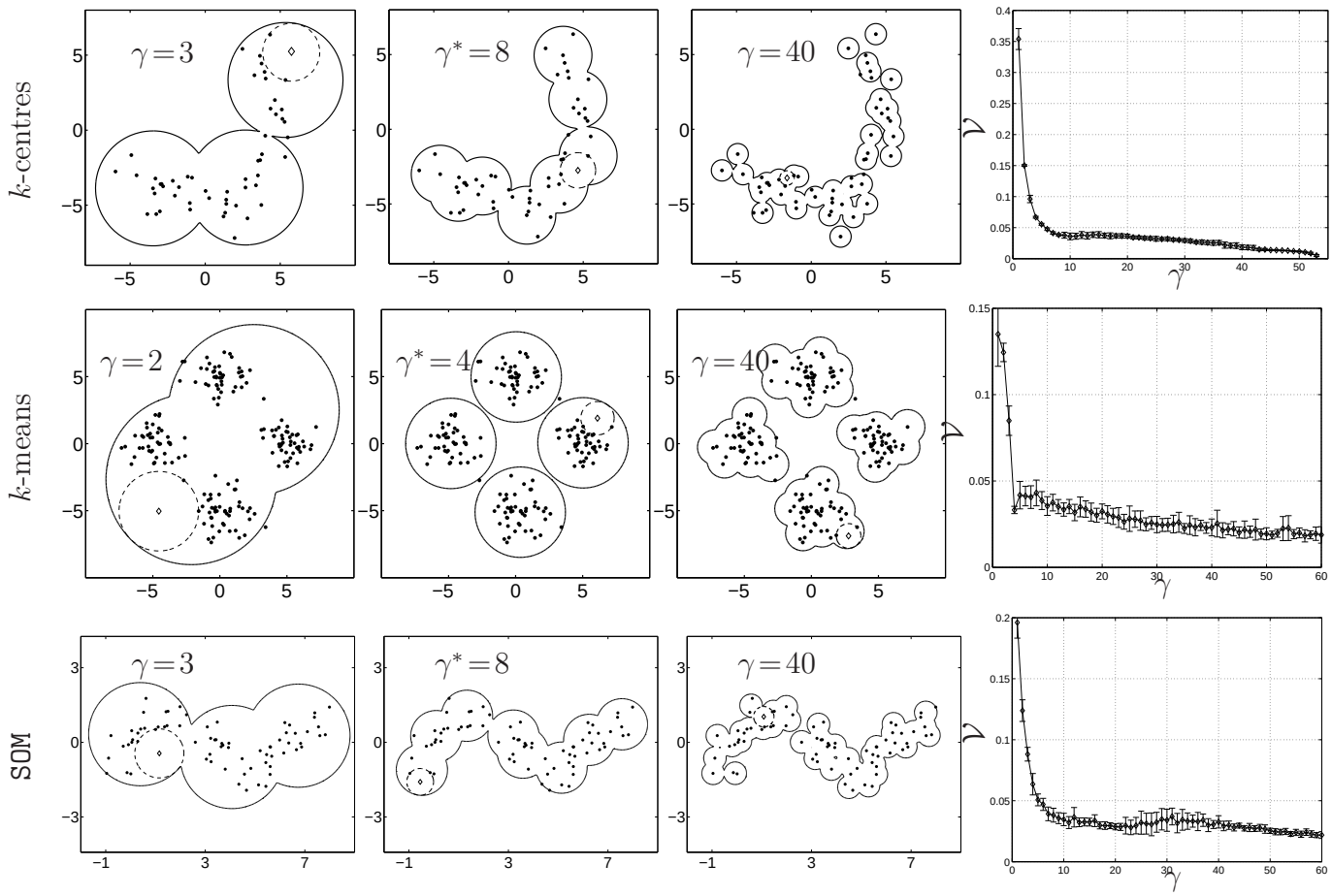


Figure 6.9: Left: Examples of 2 dimensional plots of classifiers with $\mathcal{S}(\mathbf{a}, r)$, $\varepsilon_t^{tr} = 0.01$. Right: $\mathcal{V}$-statistic for different values of γ.

In figure 6.8 when we increase the complexity of a classifier (from left to right) for k-centres for γ between 7 and 34 the complexity of a decision boundary almost does not change. Simply the same decision boundary is described by more complex classifiers. The classifiers like k-centres, k-means, SOM hardly overfit, therefore we do not observe a minimum in the value of the $\mathcal{V}$-statistic. For spherical one-class classifiers we propose to select the least complex classifier for which two volumes, V_h and $V_{\mathcal{S}(\mathbf{a},r)}$ stabilise. This value represents the least complex classifier to describe the given dataset, following the Occam's razor principle. We select the most simple classifier that describes data accurately.

6.1.3 Estimation of the largest empty *N*-sphere inside kernel-based classifiers

The natural extension of the proposed Algorithm 6.1 would be to apply it to kernel-based one-class classifiers such as SVDD [Tax and Duin, 1999] since this model is also based on a single sphere. However, in this section we show that this is not possible. We show that neither the volume nor the radius of the largest empty N-sphere as well as the volume of a classifier computed in the kernel space has a quantitative meaning in the input space since they are computed over a part of the Hilbert space $\mathcal{H}$ that has no preimage. Note that we are arguing that the volume of a sphere computed in $\mathcal{H}$ has no quantitative meaning, but still the minimisation of the volume does, similar as in SVDD and *oc*-SVM. The minimisation of the volume in $\mathcal{H}$ implies also the minimisation of the volume in $\mathbb{R}^N$.

The SVDD with a linear kernel is equivalent to the smallest N-sphere enclosing all target objects in the input space $\mathbb{R}^N$. Here, we are interested in more flexible formulations incorporated by the use of nonlinear positive definite kernels. We restrict our discussion to the Gaussian kernel $K(\mathbf{x}, \mathbf{y}) = e^{-||\mathbf{x}-\mathbf{y}||^2/\sigma^2 2}$. The SVDD is then defined as the smallest sphere enclosing all target objects, as mapped to a high- or infinite-dimensional Hilbert space $\mathcal{H}$ induced by the kernel K. Thanks to the reproducing property of K[3], the mapping does not need to be performed directly. This follows as all computations rely on inner products in $\mathcal{H}$, which are equivalent to kernel evaluations. $K(\mathbf{x}, \cdot)$ is the representation of $\mathbf{x}$ in $\mathcal{H}$. The SVDD is found as a solution to a particular quadratic optimisation problem [Tax and Duin, 1999]:

$$\begin{aligned} \max_{\alpha_i} \quad & L = \sum_i \alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \sum_i \alpha_i = 1, \\ & 0 \leq \alpha_i \leq C, \quad \forall\, i. \end{aligned} \tag{6.20}$$

where $K(\mathbf{x}_i, \mathbf{x}_j)$ expresses the inner product between the representations $K(\mathbf{x}_i, \cdot)$ and $K(\mathbf{x}_j, \cdot)$ in the space $\mathcal{H}$.

The parameters of the sphere $\mathcal{S}(\mathbf{A}_\mathcal{H}, R_\mathcal{H})$, with the smallest radius in $\mathcal{H}$, are expressed in terms of training objects and optimised weights α:

[2]Since SVDD and *oc*-SVM are equivalent when a Gaussian kernel is used the discussion although based on SVDD is also valid for *oc*-SVM.

[3]The reproducing property of the kernel K means that $\langle K(\mathbf{x}_i, \cdot), K(\mathbf{x}_j, \cdot) \rangle_\mathcal{H} = K(\mathbf{x}_i, \mathbf{x}_j)$.

$$\mathbf{A}_{\mathcal{H}} = \sum_i \alpha_i K(\mathbf{sv}_i, \cdot)$$
$$R_{\mathcal{H}} = \|\mathbf{A}_{\mathcal{H}} - K(\mathbf{sv}_i, \cdot)\|_{\mathcal{H}}$$
$$= \sqrt{1 - 2\sum_j \alpha_j K(\mathbf{sv}_i, \mathbf{sv}_j) + \sum_i \sum_j \alpha_i \alpha_j K(\mathbf{sv}_i, \mathbf{sv}_j)} \tag{6.21}$$

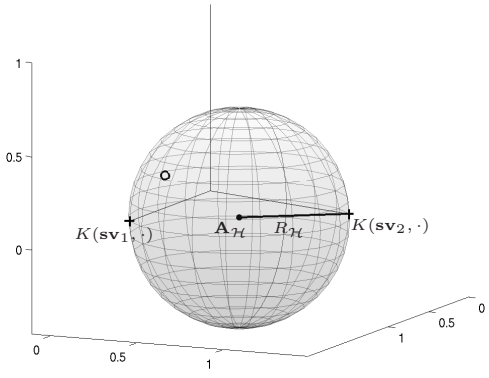


Figure 6.10: SVDD in the Hilbert space.

where $\mathbf{sv}_i$ are the support vectors, i.e. the training objects $\mathbf{x}_i$ for which the optimised α_i are positive. As the support vectors lie on the surface of the sphere $\mathcal{S}(\mathbf{A}_{\mathcal{H}}, R_{\mathcal{H}})$, any of them can be used to compute $R_{\mathcal{H}}$. Note that if the reproducing kernel map $\mathbf{x}_i \to K(\mathbf{x}_i, \cdot)$ is unknown, the centre of the SVDD cannot be determined in $\mathbb{R}^N$ since the preimage of $\mathbf{A}_{\mathcal{H}}$ does not exist in general. However, a distance of an object $\mathbf{z}_k$ to the centre can always be computed as it relies on inner products, hence kernel evaluations only.

$$d(\mathbf{z}_k, \mathbf{A}_{\mathcal{H}}) = \sqrt{1 - 2\sum_i \alpha_i K(\mathbf{z}_k, \mathbf{sv}_i) + \sum_i \sum_j \alpha_i \alpha_j K(\mathbf{sv}_i, \mathbf{sv}_j)}. \tag{6.22}$$

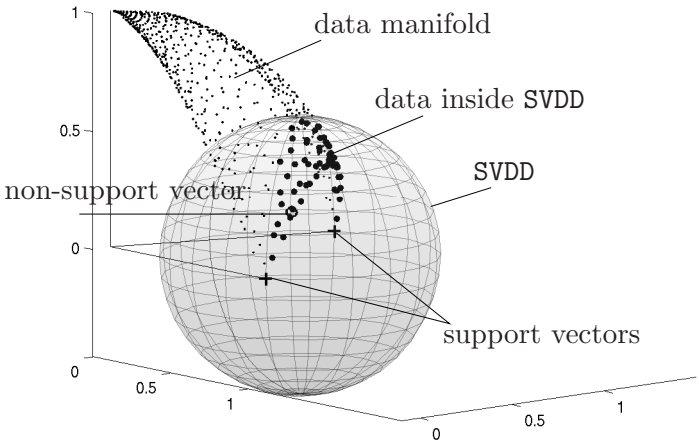
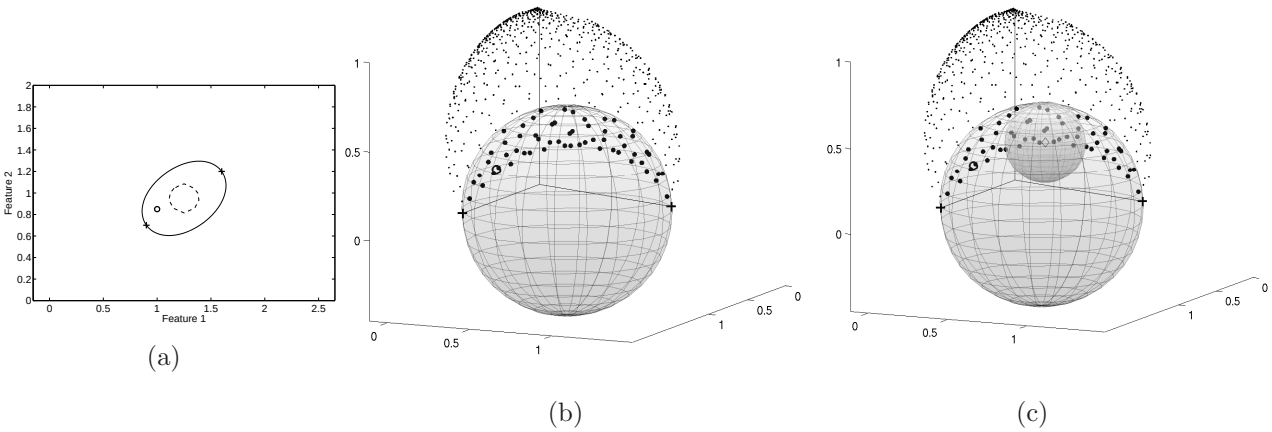


Figure 6.11: SVDD in Hilbert space.

Figure 6.10 and figure 6.11 show an example of the SVDD defined by a Gaussian kernel. {+}-s indicate two support objects and {∘} one additional nonsupport object. The angle between the support vectors axes is defined by the value of the kernel between them. Note that this figure represents the embedding to the Euclidian space of three relations between three objects in the Hilbert space. Therefore we need one extra dimension for the additional object to preserve the inner products. Since the dimensionality restriction it is only possible to picture the SVDD in $\mathcal{H}$ based on two support objects.

To find the largest empty sphere in $\mathcal{S}(\mathbf{A}_{\mathcal{H}}, R_{\mathcal{H}})$ we should modify Algorithm 6.1. In particular we can not minimise a density estimate $f(\mathbf{z})$ in $\mathcal{H}$, therefore we propose a method that is based on uniformly generated candidate centres in $\mathbb{R}^N$; see Algorithm 5.1. The candidate centres are then mapped to $\mathcal{H}$. Therefore, computation of the largest empty sphere for the



(a) (b) (c)

Figure 6.12: (a): SVDD, in $\mathbb{R}^N$, denoted by the solid line. $\{+\}$ denotes two support objects and $\{\circ\}$ one additional nonsupport objects. The dashed line denotes the largest empty sphere found in $\mathcal{H}$ with a centre on a data manifold. (b): SVDD in the Hilbert space $\mathcal{H}$ with the data manifold. (c): SVDD in the Hilbert space $\mathcal{H}$ with the data manifold and the largest empty sphere with a centre $\diamond$ on a data manifold.

non-spherical one-class classifier in $\mathbb{R}^N$ is still perform for a sphere in $\mathcal{H}$, thereby it is simpler than for an arbitrarily shaped classifier.

Assume the N-sphere $\mathcal{S}(\mathbf{A}, R)$ with minimum radius R, enclosing the training target examples X_t in the input space $\mathbb{R}^N$. We generate objects $\mathbf{z}_k$, uniformly distributed in $\mathcal{S}(\mathbf{A}, R)$ according to Algorithm 5.1 and map them to the Hilbert space $\mathcal{H}$ using kernel evaluations $K(\mathbf{z}_k, \cdot)$. Next, we determine the subset of all $\mathbf{z}_k$ that are accepted as targets by the sphere $\mathcal{S}(\mathbf{A}_\mathcal{H}, R_\mathcal{H})$. This means that $||K(\mathbf{z}_k, \cdot) - \mathbf{A}_\mathcal{H}||_\mathcal{H} \leq R_\mathcal{H}$, i.e. the representation of $\mathbf{z}_k$ in $\mathcal{H}$ is inside the sphere $\mathcal{S}(\mathbf{A}_\mathcal{H}, R_\mathcal{H})$. Figure 6.12(b) shows objects from the systematically distributed grid of the figure 6.12(a) mapped to a Hilbert space. Objects inside the SVDD sphere are marked by bold dots, and outside the SVDD sphere by smaller dots. Since training data has a finite sample size, they are placed on a manifold spanned by the support vectors. In $\mathcal{H}$ we consider only the minimum distance $d(\mathbf{z}_k, \mathbf{x})$ to a single object in the training set or the distance $d(\mathbf{z}_k, \mathbf{p}(\mathbf{z}_k))$ from $\mathbf{z}_k$ to its projection $\mathbf{p}(\mathbf{z}_k)$ onto the sphere $\mathcal{S}(\mathbf{A}_{\mathcal{H}_K}, R_\mathcal{H})$, $r_\mathcal{H} = \min[d(\mathbf{z}_k, \mathbf{x}), (d(\mathbf{z}_k, \mathbf{p}(\mathbf{z}_k)))]$. Making use of the fact that $d(\mathbf{z}_k, \mathbf{x}) = ||K(\mathbf{z}_k, \cdot) - K(\mathbf{x}, \cdot)||_\mathcal{H} = K(\mathbf{x}, \mathbf{x}) - 2K(\mathbf{x}, \mathbf{z}_k) + K(\mathbf{z}_k, \mathbf{z}_k)$ and $K(\mathbf{x}, \mathbf{x}) = 1$ for the Gaussian kernel K, after straightforward computations, we get:

$$\begin{aligned} d(\mathbf{z}_k, \mathbf{x}) &= \sqrt{2 - 2K(\mathbf{z}_k, \mathbf{x})}, \\ d(\mathbf{z}_k, \mathbf{p}(\mathbf{z}_k)) &= \sqrt{R_\mathcal{H} - d(\mathbf{z}_k, \mathbf{A}_\mathcal{H})}, \end{aligned} \tag{6.23}$$

To determine whether such sphere is inside $\mathcal{S}(\mathbf{A}_\mathcal{H}, R_\mathcal{H})$ we check:

$$r_\mathcal{H} + d(\mathbf{z}_k, \mathbf{A}_\mathcal{H}) \leq R_{\mathcal{H}_K} \tag{6.24}$$

The example of a solution to the presented algorithm is shown in figure 6.12(c). The centre of the largest empty sphere is marked by $\diamond$. As the *oc*-SVM can be reformulated into SVDD [Tax and Duin, 2001], for the Gaussian kernel, the presented discussion is also valid for this classifier, the *oc*-SVM hyperplane cuts from a kernel manifold a multidimensional sphere.

From figure 6.12(c) it can be observed that although the centre of the found sphere, as mapped from $\mathbb{R}^N$, is on the data manifold, a part of the sphere is not. This means that such part has no preimage in the input space $\mathbb{R}^N$, the lower half of the smaller sphere in figure 6.12(c). Therefore, we conclude that neither the largest empty sphere nor the sphere for which we demand that its centre is on the data manifold in $\mathcal{H}$ can be used to compute $\mathcal{V}$-statistic. Although the SVDD is always found as the smallest sphere in $\mathcal{H}$, its boundaries are also determined by the unknown data manifold. Therefore, it can happen that some part inside the found largest empty sphere has no preimage; see the common region of the two spheres without objects in figure 6.12(c).

In this section it has been shown that we can not apply the modified Algorithm 6.1 to find the largest empty sphere inside the classifier that is based on a Gaussian kernel. Although the SVDD is always found as the smallest sphere in $\mathcal{H}$, its boundaries are also determined by the unknown data manifold. Therefore, it can happen that some part of the found largest empty sphere has no preimage.

In the next section we introduce an algorithm that copes with this problem. The algorithm finds the approximately largest empty N-sphere in the input space $\mathbb{R}^N$ for non-spherical one-class classifiers including SVDD.

6.1.4 Estimation of the approximately largest empty N-sphere inside nonspherical one-class classifiers

In this section, we propose an algorithm to determine the approximately largest empty N-sphere, denoted $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$, inside a one-class classifier that is not based on a union of N-spheres. Because it is hard to determine the distance to a nonspherical decision boundary we can not use the idea of a minimisation of function (6.15). Moreover, since the decision boundary can be concave such distance does not guarantee that the entire empty N-sphere is inside a classifier. Therefore, the proposed algorithm in this section, for finding the largest empty N-sphere in a classifier h, is similar to Algorithm 5.1, based on the generation of uniformly distributed objects $\mathbf{z}_i$ in the smallest N-sphere that encloses training objects. Empty N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i)$ are determined by the distance from uniformly distributed $\mathbf{z}_i$ to their single nearest neighbours in X_t; $\rho_i = \min_{\mathbf{x} \in X_t} \|\mathbf{z}_i - \mathbf{x}\|$. To check whether an N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$ is inside or outside the classifier h, we classify a set of uniformly distributed objects generated on the surface of $\mathcal{S}(\mathbf{z}_i, \rho_i)$. To find a more accurate approximation of $\mathcal{S}(\mathbf{z}_i, \rho_i)$ we repeat the generation of uniformly distributed objects $\mathbf{z}_k$ inside each N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i)$. After the centres stabilise we choose the one with the largest radius $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$, $\tilde{r} = \max_i \rho_i$, which is our approximation of $\mathcal{S}(\mathbf{a}, r)$.

The following formula [Cramér, 1946, Muller, 1959] enables us to generate objects uniformly on the surface of the unit N-sphere $\mathcal{S}(\mathbf{0}, 1)$, centred at the origin:

$$\mathbf{s} = [s_1, s_2, \ldots, s_N] = \frac{[s'_1, s'_2, \ldots, s'_N]}{\sqrt{\sum_{i=1}^{N} (s'_i)^2}} \tag{6.25}$$

where $[s'_1, s'_2, \ldots, s'_N]$ are N independent normal deviates from $\mathcal{N}(\mathbf{0}, I)$. The points $\mathbf{s}$ are uniformly distributed on the surface of the unit N-sphere $\mathcal{S}(\mathbf{0}, 1)$. Multiplying by the radius, followed by an arbitrary translation permits the centre of the N-sphere to be different than the origin of the coordinate system. An alternative way to generate objects on the unit N-sphere is to normalise the objects generated by Algorithm 5.1 by their length, but the method presented

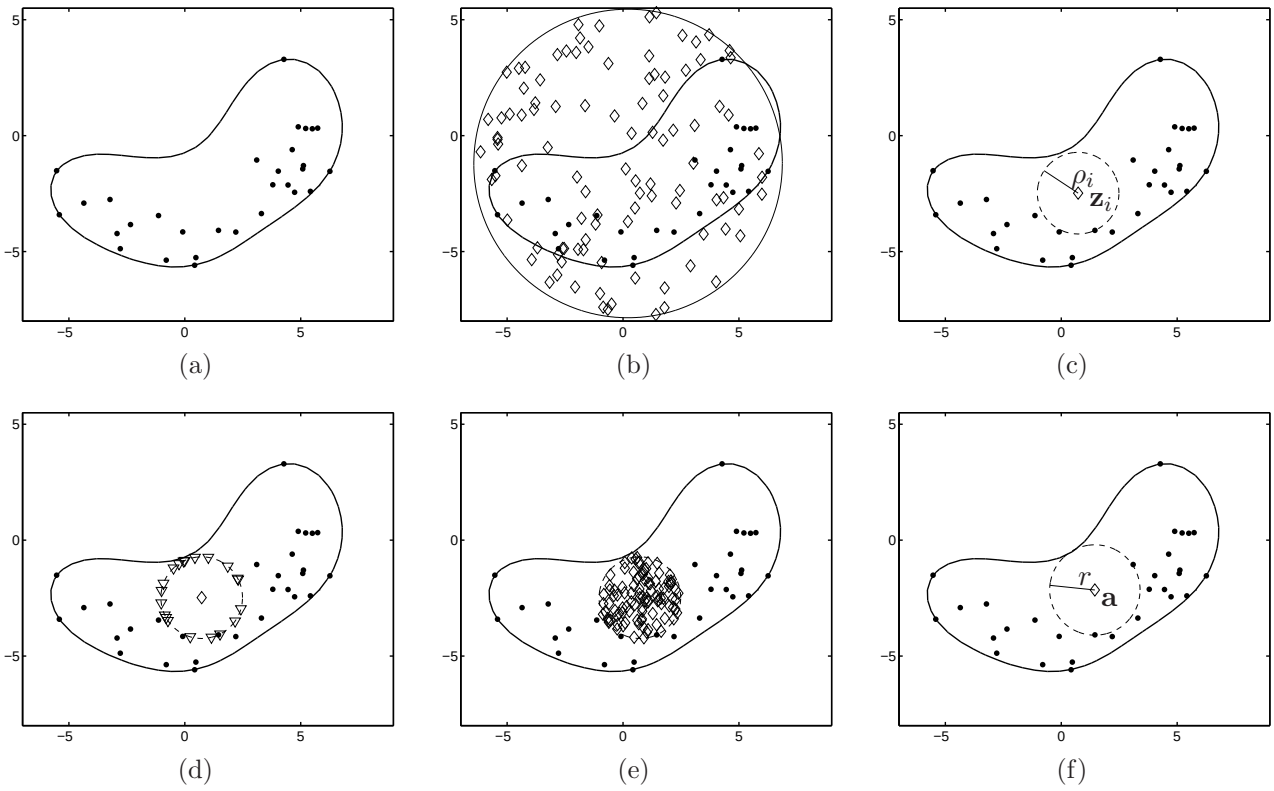


Figure 6.13: Graphical illustration of Algorithm 6.2. $\{\cdot\}$ denotes X_t, $\{\diamond\}$ uniformly distributed objects in a sphere and $\{\triangledown\}$ on the surface of a sphere. Solid line denotes a one-class classifier and dashed line denotes current estimation of $\mathcal{S}(\mathbf{a}, r)$.

here is less computationally expensive.

Assume a set of objects X_t in $\mathbb{R}^N$ and a nonspherical one-class classifier h trained on this set. Q_1, and Q_2 are parameters set by a user. Initialise $\tilde{r} = 0$.

1. Compute the smallest N-sphere that encloses the training data X_t; $\mathcal{S}(\mathbf{A}, R)$.
2. Generate Q_1 objects $\mathbf{z}_i$ uniformly distributed in $\mathcal{S}(\mathbf{A}, R)$ using Algorithm 5.1.
3. For each $\mathbf{z}_i$ find $\rho_i = \min_{\mathbf{x} \in X_t} \|\mathbf{z}_i - \mathbf{x}\|$. This determines Q_1 N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i)$.
4. To check whether an N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$ is inside the classifier h generate Q_2 uniformly distributed objects $\mathbf{s}$ on its surface according to equation (6.25) and classify all objects Q_2 by the classifier h.
5. For all $\mathcal{S}(\mathbf{z}_i, \rho_i) \subset h$ repeat steps 2-4 replacing $\mathcal{S}(\mathbf{A}, R)$ by $\mathcal{S}(\mathbf{z}_i, \rho_i)$, until $\mathbf{z}_i$ does not change.
6. Find $\tilde{r} = \max(\rho_i)$; yielding $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$.
7. Reduce $\tilde{r}$ by a constant and repeat steps 2-5, e.g. until $\tilde{r} > 0$

Algorithm 6.2: Find the largest empty N-sphere inside non-spherical classifier h.

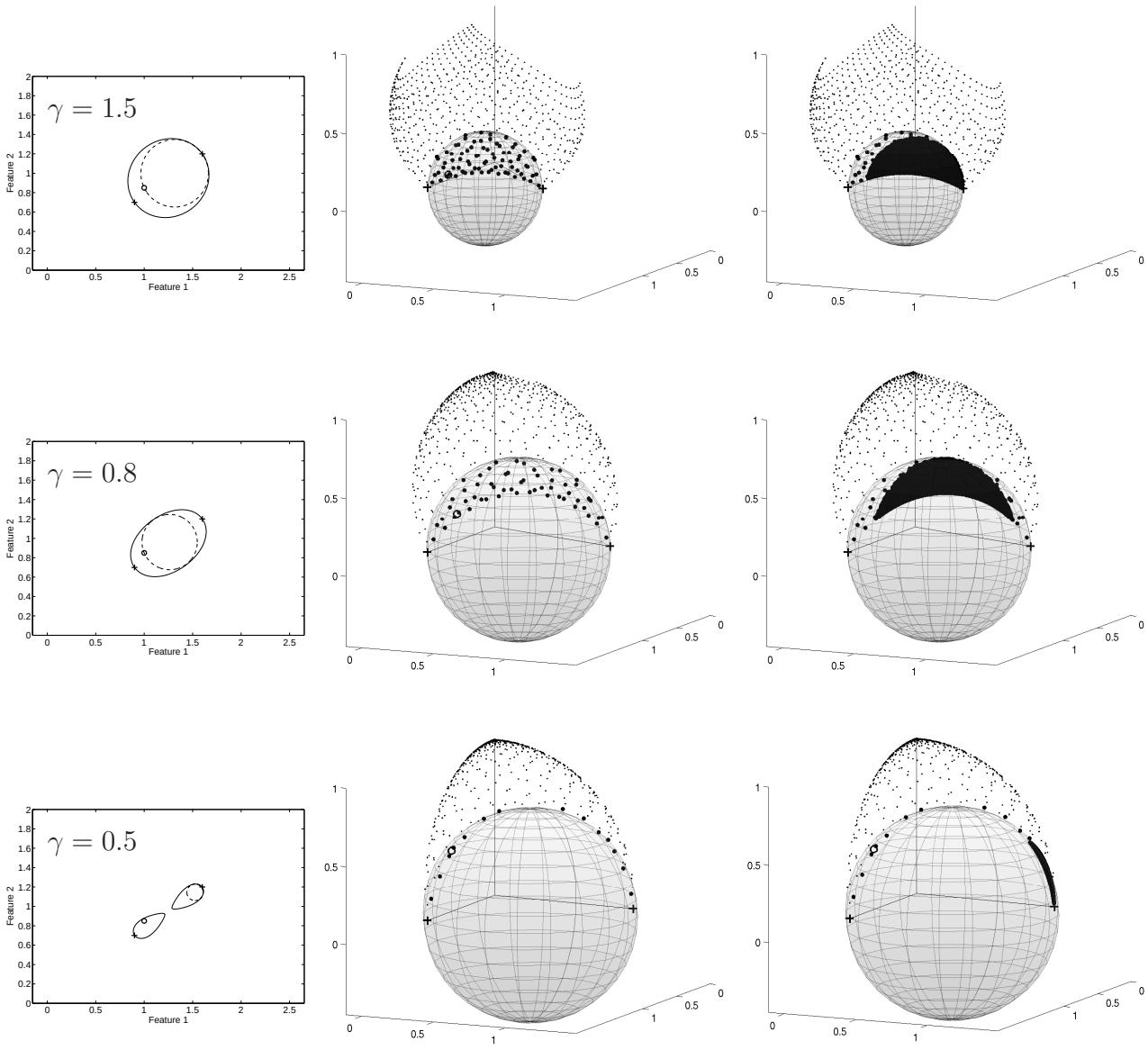


Figure 6.14: Left: SVDD description with decreasing γ from top to bottom in $\mathbb{R}^N$ with $\mathcal{S}(\mathbf{a}, r)$ (dashed line). Middle: SVDD in the Hilbert space $\mathcal{H}$. Right: SVDD in the Hilbert space $\mathcal{H}$ with the black, region represents $\mathcal{S}(\mathbf{a}, r)$ mapped to $\mathcal{H}$.

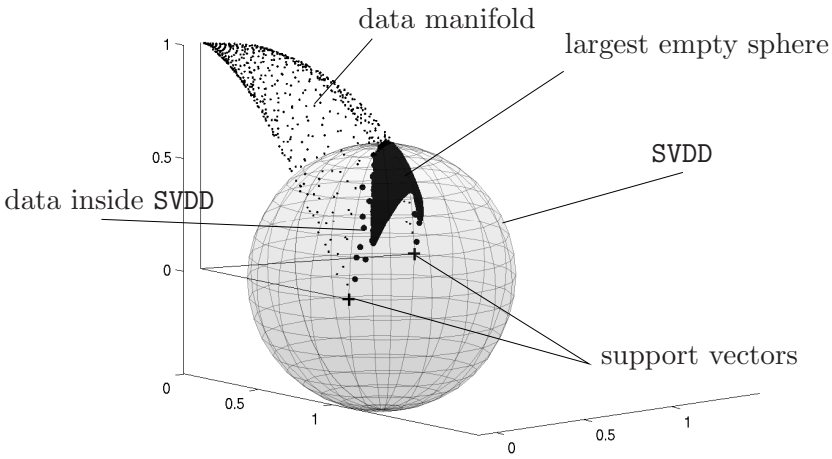
The proposed algorithm to find the approximately largest empty N-sphere $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$ inside an arbitrary one-class classifier h is summarised in Algorithm 6.2 and figure 6.13. First the smallest N-sphere that encloses all X_t is computed in the input space $\mathbb{R}^N$; figure 6.13(b). Because of the non-linearity of classifier h, part of it can be outside $\mathcal{S}(\mathbf{A}, R)$, therefore we increase R by a small constant, e.g. $R = 1.05R$. Next, a set of potential centres $\mathbf{z}_i$ of the largest empty N-sphere $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$ are generated uniformly in $\mathcal{S}(\mathbf{A}, R)$; figure 6.13(b). In step 3 a single nearest neighbour for each $\mathbf{z}_i$ is found in X_t which determines the set of empty N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i)$ where $\rho_i = \min_{\mathbf{x} \in X_t} \|\mathbf{z}_i - \mathbf{x}\|$; figure 6.13(c), here a single sphere is shown. To decide whether the N-sphere is inside or outside a classifier h we generate a set of uniformly distributed objects on the surface of an N-sphere $\mathcal{S}(\mathbf{z}_i, \rho_i)$; in figure 6.13(d). If all uniformly distributed

objects for a single $\mathcal{S}(\mathbf{z}_i, \rho_i)$, are classified by h as targets then we assume that the N-sphere is entirely inside h. For all N-spheres $\mathcal{S}(\mathbf{z}_i, \rho_i) \subset h$ steps 2-4 are repeated replacing $\mathcal{S}(\mathbf{A}, R)$ by current approximations $\mathcal{S}(\mathbf{z}_i, \rho_i)$, until $\mathbf{z}_i$ do not change; figure 6.13(e,f). Next, we determine an N-sphere with the maximum radius $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$. which is an approximation of $\mathcal{S}(\mathbf{a}, r)$. Since we use a finite set of uniformly distributed candidate centres $\mathbf{z}_i$, the obtained solution is only an approximation of the largest, empty N-sphere in h. We can improve this approximation further by repeating the above steps for decreasing $\tilde{r}$. After step 6 in Algorithm 6.2 the radius $\tilde{r}$ can be reduced by a constant, therefore for the same number of objects Q_1 and assuming $\mathbf{a} \in \mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$ $\lim_{V \to 0} \|\mathbf{a} - \tilde{\mathbf{a}}\| = 0$.

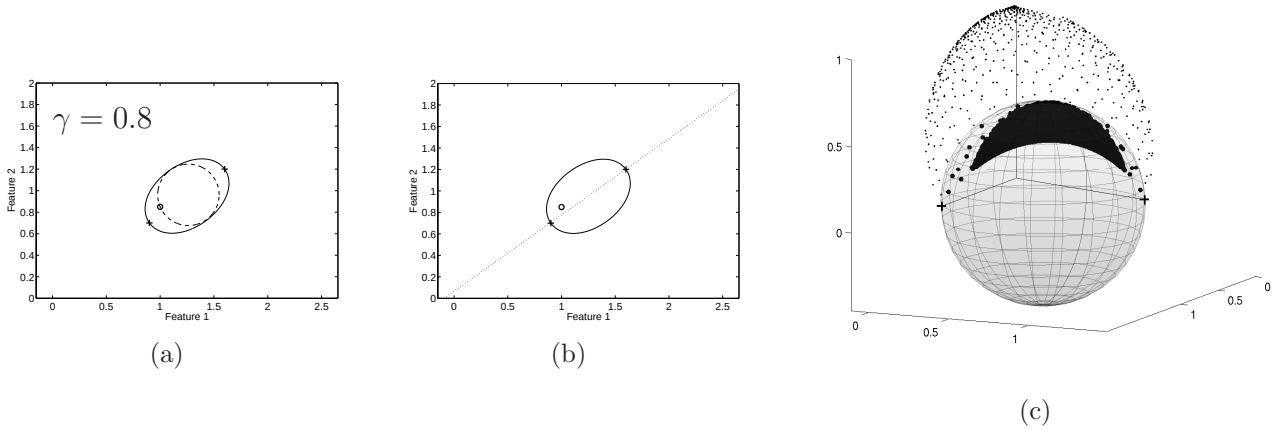
Algorithm 6.2 is relatively fast, since it performs mainly classification of objects, however, it allows to find only the approximately largest empty N-sphere. The approximation depends on the numbers Q_1, Q_2 of uniformly generated objects. We discuss the influence of the number of generated objects on the approximation factor in subsection 6.1.4.

Now, we can show how the largest empty sphere $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$ found in the input space $\mathbb{R}^N$ in SVDD which is based on a Gaussian kernel looks like in the Hilbert space $\mathcal{H}$. Figure 6.14(Left) shows the largest empty N-sphere found in SVDD in $\mathbb{R}^N$ using Algorithm 6.2 for various values of γ. The middle column of 6.14 shows the SVDD in the Hilbert space with a data manifold. The right column shows the projection of $\mathcal{S}(\tilde{\mathbf{a}}, \tilde{r})$ in $\mathcal{H}$, denoted by black patches.

The figure on the right shows relations between objects, SVDD and the largest empty sphere in the Hilbert space. The figure shows SVDD from a bit different angle than figure 6.14 to emphasise a 3D structure.



We have mapped the systematically distributed objects on the grid of the left figures to $\mathcal{H}$ using kernel evaluations. The objects approximate the data manifold in $\mathcal{H}$ shown in figure 6.14(Middle) and 6.14(Right) as dots. Objects inside the SVDD sphere are marked by bold dots and outside by smaller dots. Since, by Taylor expansion of $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-||\mathbf{x}_i - \mathbf{x}_j||^2/\gamma^2} = 1 - \frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{\gamma^2} + \ldots$, then for decreasing γ, the support vectors become nearly orthogonal in the Hilbert space as the inner product $K(\mathbf{sv}_i, \mathbf{sv}_j)$ approaches 1. This can be seen by observing the increasing radii of SVDD spheres in the middle column of figure 6.14. From the right column of figure 6.14 it can be seen that for two support objects the sphere in $\mathbb{R}^N$ has a moon like shape in $\mathcal{H}$.



(a) (b) (c)

Figure 6.15: (a): SVDD description in $\mathbb{R}^N$ with $\mathcal{S}(\mathbf{a}, r)$ (dashed line) (b): Reflection line (doted line). (c): SVDD in the kernel space $\mathcal{H}$ with the black, moon shaped area represents $\mathcal{S}(\mathbf{a}, r)$ mapped to $\mathcal{H}$.

Observation
If the number of support vectors in kernel based classifiers is lower than $N+1$, where N is the dimensionality of an input space $\mathbb{R}^N$, there is a reflection or rotation hyperplane that determines regions of $\mathbb{R}^N$ that are mapped at the same position in $\mathcal{H}$. This follows from triangulation properties, in $\mathbb{R}^N$ in order to uniquely describe the position of an arbitrary object we need $N+1$ reference objects. If the number of support vectors equals N there is a reflection hyperplane, figure 6.15.(b). If the number of support vectors is smaller than N there is a rotation hyperplane.

Note that the shorter edge of the moon shape "sphere" in the Hilbert space is the part of the segment connecting two support vectors in $\mathbb{R}^N$ inside $\mathcal{S}(\mathbf{a}, r)$. The line that goes through two support vectors is the reflection line. This means that two object which are projected onto the same point of the line can not be distinguished in a Hilbert space. Their inner product relations with support vectors are the same.

Using Algorithms 6.1 and 6.2 the $\mathcal{V}$-statistic can be computed for an arbitrary one-class classifier. As an example the $\mathcal{V}$-statistic is computed for three non-spherical one-class classifiers: a decision-based SVDD, a neural network-based auto-encoder and a density-based classifier: Parzen, see figures 6.16 and 6.17. We trained these classifiers on one of our toy examples the $2D$ banana-shaped target class. For these three classifiers γ denotes respectively sigma in a Gaussian kernel, the number of hidden units and the smoothing parameter. Figure 6.17 and 6.16 show the $\mathcal{V}$-statistic with some examples of plots of data and classifiers with different values of γ. The complexity parameter γ changes from left, the most simple, to right, the most complex classifier. For the SVDD, the auto-encoder and Parzen the $\mathcal{V}$-statistic has a minimum, here $\gamma^* = 3$ for SVDD $\gamma^* = 10$ for auto-encoder, and $\gamma^* = 1$ for Parzen. For these classifiers the $\mathcal{V}$-statistic decreases, but at a certain moment some stable patches appear that hardly change volume. Because inside these patches a small N-sphere can still be fitted, the numerator in equation (6.2) is almost constant. On the other hand the total volume of the classifier decreases, causing the denominator to decrease. This results in the $\mathcal{V}$ to increase. This can be observed in the last row of figure 6.17, which shows on the right figure the relation between V_h and $V_{\mathcal{S}(\mathbf{a},r)}$ for the Parzen classifier. We can see that for V_h smaller than 100, $V_{\mathcal{S}(\mathbf{a},r)}$

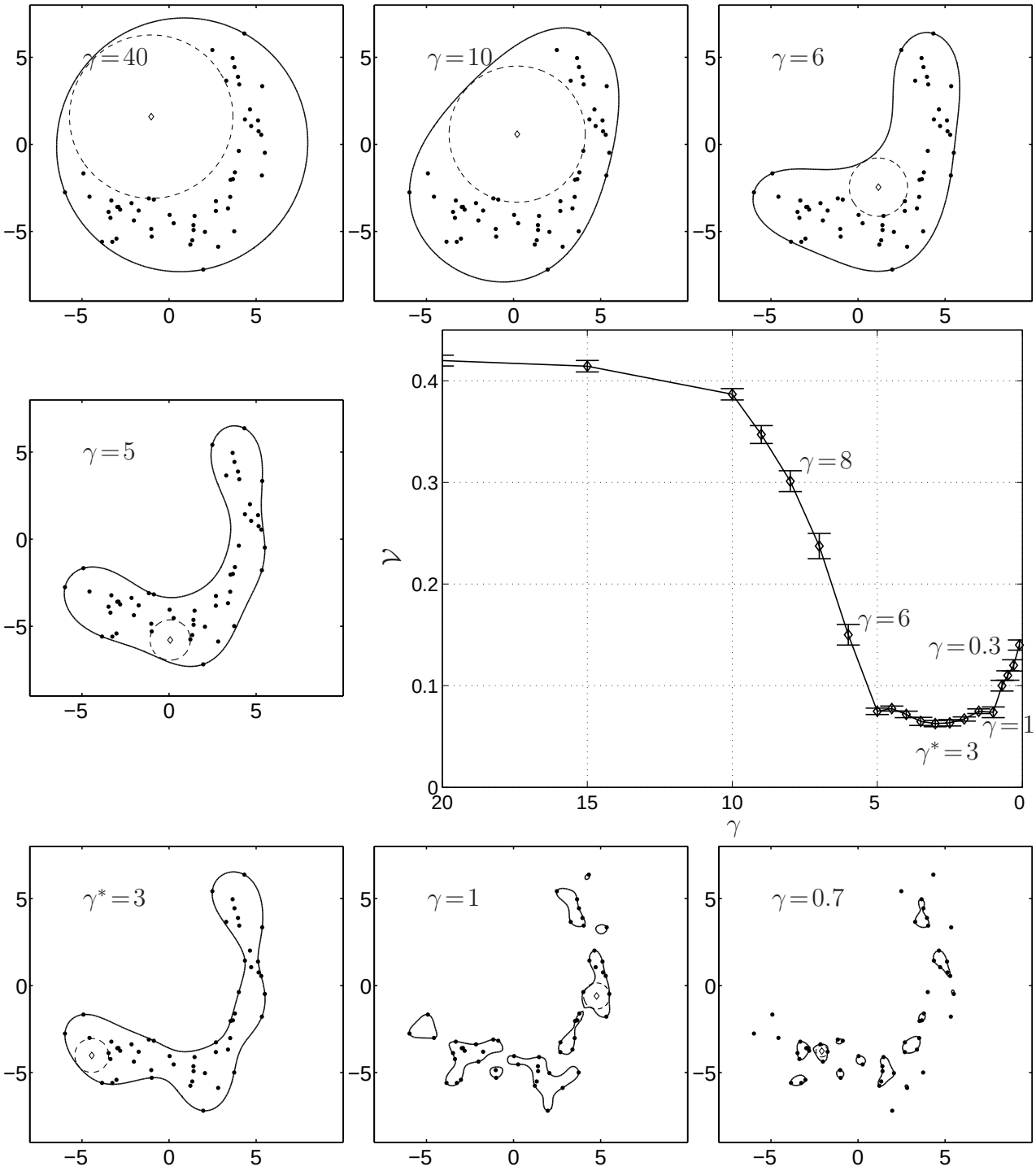


Figure 6.16: The largest empty N-sphere determined inside `SVDD` descriptions based on a Gaussian kernel for decreasing γ, $\varepsilon_t^{tr} = 0.01$. The centres of $\mathcal{S}(\mathbf{a}, r)$ are indicated by $\diamond$. The corresponding $\mathcal{V}$-statistic for different values of γ is shown on the right.

almost does not change, therefore $\mathcal{V}$ rises. Such classifiers over- and underfit therefore we can observe the minimum in the value of $\mathcal{V}$. This phenomenon does not happen when the volume of a classifier decreases in the same rate as the volume of the largest empty N-sphere e.g. for symmetrically distributed data where data can not be distinguished in terms of distances.

For $\gamma = 1$ the volume of the auto-encoder is infinite, since the classifier consists of two

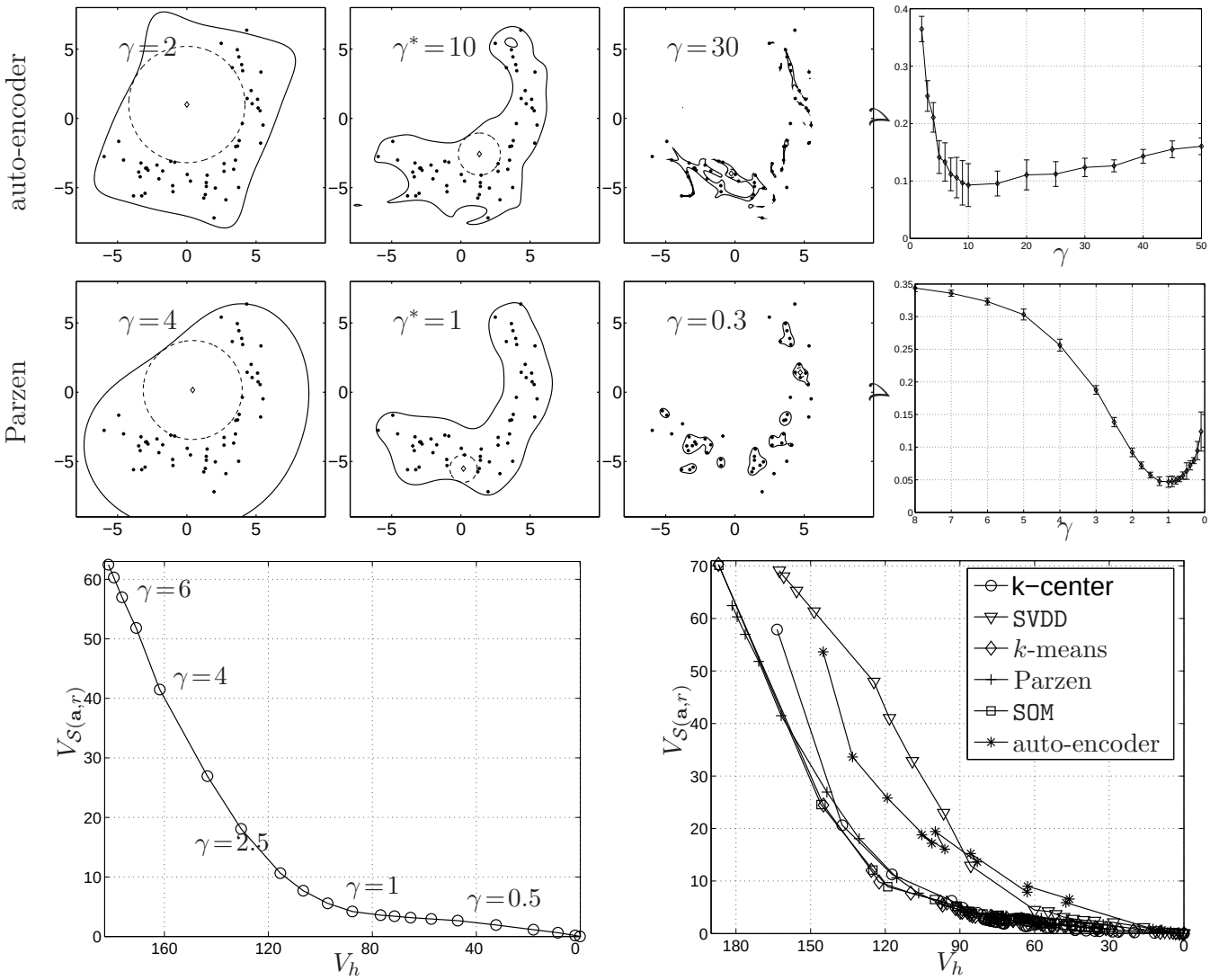


Figure 6.17: $\mathcal{V}$-statistic computed for auto-encoder and Parzen classifier with examples of classifiers trained with different value of γ. In the bottom row the figure on the left shows relation between two volumes for the Parzen classifier and figure on the right relation between two volumes for all classifier trained on banana-shaped data.

parallel hyperplanes, therefore the first value of $\mathcal{V}$ has been plotted for $\gamma = 2$.

The last subfigure in 6.17 shows relation between V_h and $V_{\mathcal{S}(\mathbf{a},r)}$ for all classifier that we have trained on $2D$ banana-shaped class. Such a figure, similar to a ROC curve, can be used to select a classifier that fits the data best for all complexity parameters. The classifier with the minimum area under the $\mathcal{V}$ curve may be the classifier that best fits data for an analysed set of complexity parameters according to $\mathcal{V}$-statistic.

Since there is not a single characteristic shape of the $\mathcal{V}$-statistic curve, we propose two different selection criteria for γ. For classifiers like k-means, k-centres, `SOM` and also for mixture of Gaussians the optimal γ is selected based on a stability measure, see figure 6.9. For `SVDD`, Parzen and auto-encoder we select $\gamma*$ for which $\mathcal{V}$-statistic has the minimum value, see figures 6.16-6.17. We can also unify this criterion by saying that we select the most complex classifier that does not overfit, which is indicated by clear increase in the value of the $\mathcal{V}$-statistic.

Approximation factor

The solutions given by Algorithm 6.2 depend on various parameters like the data dimensionality N, the number of generated candidate centres Q_1 etc. In this section, we provide explanation how the selection of these parameters influence the accuracy of a final solution.

The probability that an uniformly distributed object generated in an N-sphere $\mathcal{S}(\mathbf{a}_1, r_1)$ is also in an N-sphere $\mathcal{S}(\mathbf{a}_2, r_2)$, where $\mathcal{S}(\mathbf{a}_2, r_2) \subset \mathcal{S}(\mathbf{a}_1, r_1)$, is equal to the ratio of two volumes $\frac{V_{\mathcal{S}(\mathbf{a}_2, r_2)}}{V_{\mathcal{S}(\mathbf{a}_1, r_1)}} = (\frac{r_2}{r_1})^N$. Therefore, the probability that a single object, from such a generated uniform distribution, is not in the N-sphere $\mathcal{S}(\mathbf{a}_2, r_2)$ is $1 - (\frac{r_2}{r_1})^N$. Since we generate Q_1 such objects t times in $\mathcal{S}(\mathbf{a}_1, r_1)$, the probability that none of these objects are in $\mathcal{S}(\mathbf{a}_2, r_2)$ is $(1 - (\frac{r_2}{r_1})^N)^{tQ_1}$. This probability can be bounded from above:

$$(1 - (\frac{r_2}{r_1})^N)^{tQ_1} \leq e^{-(\frac{r_2}{r_1})^N tQ_1}$$

where we used the property that $(1 - \mathbf{x})^{\mathbf{y}} \leq e^{-\mathbf{xy}}$. This is the probability that none of tQ_1 uniformly generated objects in $\mathcal{S}(\mathbf{a}_1, r_1)$ are in $\mathcal{S}(\mathbf{a}_2, r_2)$. We would like to use this result to determine the number of objects required to reduce this probability of failure below some desired level δ.

$$e^{-(\frac{r_2}{r_1})^N tQ_1} \leq \delta$$

Rearranging terms to solve for Q_1, we find:

$$Q_1 \geq \left(\frac{r_1}{r_2}\right)^N \frac{1}{t} \ln(\frac{1}{\delta}) \tag{6.26}$$

As an example; if we would like to find an N-sphere $\mathcal{S}(\mathbf{a}_2, r_2)$ with an accuracy equal to half of the current radius r_1, in $N = 15$, with probability of failure $\delta = 0.1$ and for $t = 20$, Q_1 should be $Q_1 \geq 3800$ and for $N = 20$, Q_1 should be $Q_1 \geq 120,000$. In our experiments we used $Q_1 = 100,000$ which is quite close to this number.

In Algorithm 6.2, we check whether an N-sphere $\mathcal{S}(\mathbf{z}, \rho)$ is inside or outside the classifier by classifying Q_2 uniformly distributed objects $\mathbf{s}$ on the surface of $\mathcal{S}(\mathbf{z}, \rho)$. Under the assumption that the classifier is convex between generated objects $\mathbf{s}$, the maximum volume of an N-sphere $\mathcal{S}(\mathbf{z}, \rho)$ that can be outside a classifier for the N-sphere, that all generated objects $\mathbf{s}$ are inside a classifier h can be computed from the formula of the volume of a spherical cap (6.6). Using inequality (6.26) the radius r_c and height h_c of such a spherical cap can be determined as $(r_c = r_2;\ \rho = r_1)$:

$$r_c \geq \rho \sqrt[N-1]{\frac{\ln(\frac{1}{\delta})}{Q_2 t}}, \qquad h_c = \rho - \sqrt{\rho^2 - r_c^2} \geq \rho\left(1 - \sqrt{1 - \sqrt[N-1]{\left(\frac{\ln(\frac{1}{\delta})}{Q_2 t}\right)^2}}\right)$$

h_c was determined using simply the Pythagoras equation to two triangles shown in figure B.1. Therefore, with probability δ none of $Q_2 t$ uniformly distributed objects on the surface of $\mathcal{S}(\mathbf{z}, \rho)$ is in an $N-1$ dimensional sphere with radius larger than r_c. Substituting the above inequalities into equation (6.6) the ratio between the volume of the entire N-sphere $\mathcal{S}(\mathbf{z}, \rho)$

and the maximum volume that can be outside the classifier is:

$$\frac{V_{cap}}{V_{\mathcal{S}(\mathbf{z},\rho)}} \geq \frac{C}{\rho\sqrt{\pi}} \int_0^{\beta_{max}} \sin^{N-1}(\beta)d\beta \quad \text{where} \quad \beta_{max} = \arcsin(\sqrt{(2\rho - h_c)(h_c/\rho^2)}) \tag{6.27}$$
$$C = \frac{\Gamma(N/2+1)}{\Gamma((N-1)/2+1)}.$$

Under the assumption that a classifier is not convex between objects $\mathbf{s}$ the part of the volume of the N-sphere that is considered inside a classifier and can be outside it is divided by the volume of such N-sphere, is bounded by the above inequality with the probability δ. As an example; for $\delta = 0.1$, $N = 15$, $Q_2 = 10,000$, $t = 20$ and $\rho = 1$ above ratio equals $3 \cdot 10^{-7}$ and for $N = 20$, equals $4 \cdot 10^{-7}$. Such an error can be neglected in our computations.

6.2 Experiments

In this section we apply the algorithms 6.1 and 6.2 to datasets from the UCI repository [Hettich et al., 1998] and to a machine condition monitoring dataset [Ypma, 2001]. We compare the performance of the models selected by the $\mathcal{V}$-statistic with standard model selection methods based on the uniformly generated outliers and the method based on the consistency of a one-class classifier.

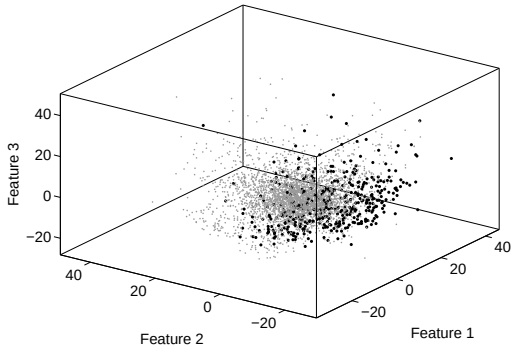


Figure 6.18: The PCA projection of *Concordia* dataset. The target objects of the digit {4} are marked by bold dots.

First, we focus on the prediction of γ, based on the $\mathcal{V}$-statistic for a particular type of classifier. The *Concordia* hand-written digits dataset has been chosen with digit {4} as the target class. The objects from the other classes: $\{0:9\}\backslash\{4\}$, have been labelled as outliers. One can expect not only outlier objects that belong to the class of written digits but also e.g. letters, special symbols like an exclamation mark or objects that do not represent recognisable classes. Since we do not have examples of these classes represented in data, similar to [Tax and Duin, 2001] we assume that these classes are uniformly distributed and by generating artificial data we would like to model appearance of such classes. Therefore, in addition to outliers constituted from digits we generated the same number of uniformly distributed outliers, using Algorithm 5.1.

The dataset was mapped from 256 to 15 dimensions preserving about 85% of the variance in the target class. The dataset mapped on the first three PCA dimensions of class {4} is shown

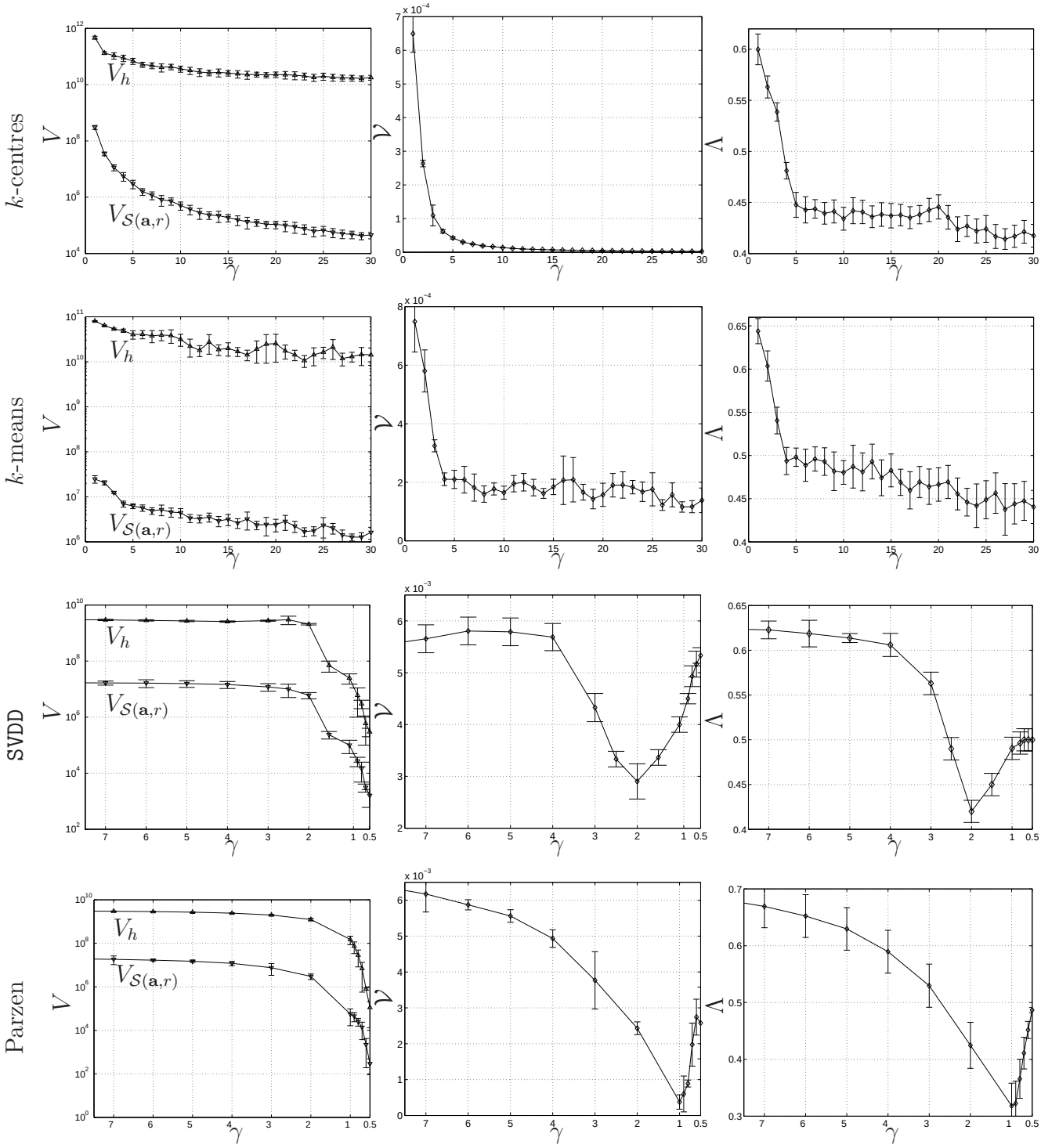


Figure 6.19: Left: The volume, in the logarithmic scale, of one-class classifiers trained on digit {4} from the *Concordia* dataset. The results were averaged over 20 random splits of the data. Middle: The corresponding $\mathcal{V}$-statistic. Right: The classification error.

in figure 6.18. Objects from the target class are marked by bold dots and the outlier objects, from classes $\{0:9\}\backslash\{4\}$ by small dots. Since the target class surrounds the outlier class the estimation of γ is especially important for the classification performance. To compute the $\mathcal{V}$-statistic we use Algorithm 6.1 and equation (6.10) for k-centres and k-means. For `SVDD` and

Parzen we use Algorithm 5.1 and Algorithm 6.2 to compute the $\mathcal{V}$-statistic. The results are shown in figure 6.19. The first column shows the computed volume V_h of a classifier together with the volume of the largest empty N-sphere $\mathcal{S}(\mathbf{a}, r)$ for different value of γ. The complexity of classifiers increases from left to right. The middle column shows the corresponding $\mathcal{V}$-statistic. The last column shows the classification error Λ on the test set with $\lambda = 0.5$; see equation (5.2). The experiments were averaged over 20 random splits of the data into training and test sets. The number of uniformly generated candidate centres in Algorithm 6.2 was set to $100,000$. Based on the $\mathcal{V}$-statistic complexity parameters can easily be determined by selecting either the minimum for Parzen and `SVDD` or the stability point for k-centres and k-means. In figure 6.19 the $\mathcal{V}$-statistic for the `SVDD` classifier has a local minimum around $\gamma = 7$, which means the volume of the classifier decreases faster than the volume of $\mathcal{S}(\mathbf{a}, r)$, e.g. $\mathcal{S}(\mathbf{a}, r)$ is determined only by objects from the training set X_t.

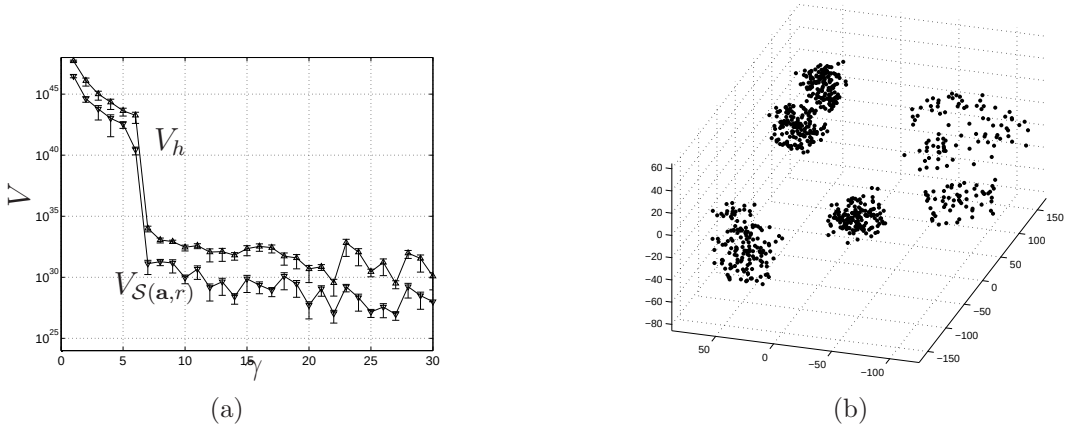


Figure 6.20: (a): The volume of k-centres classifier V_h trained on the Delft pump 5×3 dataset and the volume of the largest empty N-sphere inside h. (b): The projection of the target class on the first three `PCA` components.

A similar problem of determining the complexity of a chosen classifier is shown in figure 6.20. Here, we would like to present a negative result, where we can not select a model based on the $\mathcal{V}$-statistic. The problem is the 64 dimensional machine diagnostic problem selected from the Delft pump dataset [Ypma, 2001]. The dataset contains measurements of normal and abnormal working stages of a water pump. Due to the data dimensionality the $\mathcal{V}$-statistic can only be determined for one-class classifiers consisting of N-spheres. Because for a limited number of generated objects ($\sim 100,000$) all objects lie on the surface of the smallest N-sphere that encloses X_t. For the same reason the outlier class has not been enriched with uniformly generated objects. To solve this classification problem k-centres has been chosen, as it does not suffer from the curse of dimensionality.

Figure 6.20(a) shows the volume of a k-centre classifier, V_h, and the largest empty N-sphere, $V_{\mathcal{S}(\mathbf{a},r)}$ for different values of the complexity parameter. The target class is constructed from 752 examples of a a normally working water pump. The computation of the volumes was averaged over 20 trials of 50% randomly chosen objects from the target class.

We can see that the values of the two volumes are noise (see the scale of V). Therefore, dividing these to values gives also non-smooth characteristic of the $\mathcal{V}$-statistic, from which we can not automatically select a model. The reason of the non-smooth characteristic of $\mathcal{V}$ is that in 64 dimensions all distances between objects are large. The space is empty because for

the given problem there is not enough data to create any shape of a target class. Therefore, almost all complexity parameters are equally good according to the $\mathcal{V}$-statistic.

However, we can clearly see, from the left figure 6.20, that there is a sudden decrease in the value of the two volumes, between $\gamma = 6$ and $\gamma = 7$. For $\gamma \geq 7$ the two volumes do not change much. In the figure 6.20 we show the target class projected on the first three PCA components. We can see 7 clusters. Since the data has been projected from 64 to 3 dimensional space we can not guarantee that this data structure is also in the original space. However, assuming that such structure also exist in 64 dimensions we can select the complexity parameter as $\gamma = 7$, based only on the volume of the classifier. Namely, for $\gamma \geq 7$ the shape of a classifier does not change.

From the above example we can see that in small sample size problems the selection of complexity parameters can be arbitrary and can require prior knowledge about a recognition problem.

6.2.1 Comparison with other model selection methods

In this subsection we compare the performance of models selected according to the $\mathcal{V}$-statistic with the performance of models selected by uniformly generated outliers [Tax and Duin, 2001] and the consistency model selection [Tax and Müller, 2004]. The comparison is made on two classifiers, k-centres and Parzen for 13 UCI datasets. The results are summarised in table 6.1 and 6.2. For all 13 dataset the first class has been chosen as the target class and all other classes constituted the outlier class. To determine γ using Algorithm 6.2 for the Parzen classifier for datasets with dimensionality larger than 20 were mapped on the first 15 PCA directions of the training set. In datasets up to 20 dimensions the outlier class was enriched with the same number, as the size of the genuine outlier class, of uniformly generated outliers. For k-centres the parameter γ has been selected from $\gamma = [1, \ldots, 30]$ and for Parzen γ has been selected from a range of 30 numbers equally spaced between the minimum and the maximum distance between objects in the training set. The threshold θ was set to $\varepsilon_t^{tr} = 0.05$. The averaged classification error was obtained using 10×3 fold cross-validation.

From the tables 6.1 and 6.2 one can see that the performance of the proposed model selection criterion is either similar to the compared criteria or it outperforms them. One of the reasons that the proposed method based on the $\mathcal{V}$-statistic performs better than the method based on the uniform outlier generation or the consistency method in the presented experiments, is the evaluation criterion used by us. In all experiments, we combine uniformly generated outliers with the existing, outlier classes present in the dataset. We do so to represent the occurrence of classes in real applications. One can expect that, for example, in the handwritten digit {4} recognition problem, the probability that an outlier object is from an other digit class is much higher than from the non-digit outlier class. Although we still would like to predict the error on the non-digit outlier class by combining given and uniform distributed outliers in the test set. The other reason is that the UCI repository datasets are two- or multi-class problems. When they are transformed to a one-class classification problem, they overlap only in certain regions of the feature space and do not cover the entire region where the target class is. Therefore, minimising the largest empty N-sphere in a classifier better represents such problems than the minimisation of the volume of an entire classifier as in the consistency or the uniform outlier generation methods. Moreover, for the method based on uniformly distributed outliers in high dimensional spaces, non or a few uniformly generated outlier objects are classified as targets. Consequently, the model selection is based on the error

Table 6.1: The complexity parameter γ for k-centres classifier optimised by the uniform outlier generation method γ_u, the consistency approach γ_c and the $\mathcal{V}$-statistic $\gamma_\mathcal{V}$ with corresponding classification error Λ. $\#t/o$ denotes size of target and outlier class.

dataset	N	#t/o	γ_u	γ_c	$\gamma_\mathcal{V}$	Λ_u	Λ_c	$\Lambda_\mathcal{V}$
biomed	5	172/134	4.4(2.2)	3.3(0.8)	17.4(8.0)	13.9(2.4)	15.1(1.5)	**9.2(0.7)**
liver	6	145/400	4.7(2.6)	4.2(1.9)	18.1(9.2)	26.9(1.3)	26.9(2.0)	**22.6(2.7)**
mfeat-mor	6	200/3600	3.1(1.5)	7.6(4.6)	7.1(1.3)	6.1(1.6)	6.1(2.4)	6.1(1.1)
ecoli	7	143/258	9.1(2.1)	3.5(1.0)	9.8(1.1)	11.7(3.2)	13.1(3.0)	11.1(3.7)
diabetes	8	500/536	3.0(1.1)	8.2(4.4)	10.0(3.4)	23.3(0.9)	23.7(0.8)	**11.8(0.9)**
breast	9	444/478	8.4(4.7)	6.8(2.8)	10.5(3.2)	4.5(1.3)	5.2(0.8)	4.1(2.0)
heart	13	160/274	2.8(1.4)	4.5(1.5)	16.7(4.9)	25.1(1.3)	25.5(1.3)	**11.2(4.2)**
waveform	21	1657/6686	7.9(4.7)	10.3(3.4)	15.1(3.0)	14.8(1.3)	14.7(1.3)	15.9(2.1)
ionosphere	34	225/252	4.2(2.4)	4.6(1.6)	7.4(1.5)	7.8(3.2)	6.9(1.5)	5.7(1.9)
satellite	36	1533/9804	5.1(5.1)	12.8(4.7)	9.2(2.1)	7.4(2.7)	**5.3(1.5)**	7.1(1.2)
mfeat-zer	47	200/3600	2.8(1.5)	4.2(1.4)	12.4(2.1)	6.2(1.8)	4.9(1.0)	3.9(2.1)

on the target class only. Therefore the selection is biased to classifiers with larger volumes.

Compared to the consistency method Algorithm 6.2 can only be applied in low dimensional problems, up to $20D$, for non-spherical one-class classifiers, however the consistency method does not depend on the dimensionality of data.

Table 6.2: The complexity parameter γ for Parzen classifier optimised by the uniform outlier generation method γ_u, the consistency approach γ_c and the $\mathcal{V}$-statistic $\gamma_\mathcal{V}$ with corresponding classification error Λ. $\#t/o$ denotes size of target and outlier class.

dataset	N	#t/o	γ_u	γ_c	$\gamma_\mathcal{V}$	Λ_u	Λ_c	$\Lambda_\mathcal{V}$
biomed	5	172/134	23.7(1.2)	14.4(8.6)	25.3(0.4)	10.9(1.2)	26.8(16.5)	11.3(1.0)
liver	6	145/400	22.5(7.6)	16.7(4.7)	19.9(0.5)	22.6(1.9)	29.5(6.8)	20.2(6.7)
mfeat-mor	6	200/3600	3447.4(3572.2)	0.5(0.1)	841(54)	4.9(1.5)	50.0(0)	4.5(2.3)
ecoli	7	143/258	0.1(0.02)	0.09(0.02)	0.1(0.05)	9.6(2.3)	13.1(9.6)	9.6(1.1)
diabetes	8	500/536	15.1(4.3)	4.4(0.7)	57.0(19.8)	36.5(10.3)	49.1(0.9)	**24.0(0.3)**
breast	9	444/478	2.0(0.08)	1.0(0)	1.0(0)	3.8(1.0)	6.3(0.9)	6.3(1.0)
heart	13	160/274	9.9(4.4)	6.9(4.1)	48.9(15.1)	44.2(7.1)	47.2(7.8)	**25.9(1.0)**
waveform	21/15	1657/6686	4.6(0.6)	1.8(0.2)	2.5(0.1)	21.1(2.1)	17.5(6.5)	**14.5(0.5)**
ionosphere	34/15	225/252	1.5(0.2)	1.2(0.4)	1.5(0.4)	17.3(2.7)	**14.8(2.6)**	17.1(2.1)
satellite	36/15	1533/9804	11.2(1.3)	7.1(0.5)	46.8(11.0)	10.3(2.1)	36.9(2.9)	**6.0(2.6)**
mfeat-zer	47/15	200/3600	124(13)	75.9(16.3)	183.4(32.6)	13.1(9.5)	20.8(13.7)	**4.1(0.9)**
sonar	60/15	97/222	0.8(0.09)	0.9(0.3)	1.1(0.3)	30.8(3.5)	31.8(4.2)	29.6(3.7)
concordia	256/15	400/7200	2.1(0.2)	1.8(0.3)	1.4(0.3)	21.8(4.9)	14.7(5.3)	**7.4(0.8)**

6.3 Conclusions

One of possible definitions of an outlier class is that the class contains objects which are remote from the bulk of the target class. Two fundamental questions should be asked: how remote the object should be to be considered an outlier and how do we describe the bulk of data. In this chapter, we have used one-class classification methods to describe the bulk of the data and we focus on the optimisation of complexity parameters of such methods or alternatively on the selection between several classification methods. Since during training, outlier objects

are not available we can not estimate the error on the outlier class without making additional assumptions about its distribution.

In this chapter we have proposed a model selection criterion for one-class classifiers called the $\mathcal{V}$-statistic. The criterion is based on two volumes: the volume of a one-class classifier and the volume of the largest empty N-sphere that can be found inside the classifier. For a larger volume of the one-class classifier, we expect a smaller target rejection rate and larger outlier acceptance rate. On the other hand, for a smaller volume of the classifier we expect a larger target rejection rate and smaller outlier acceptance rate. To define the correct volume of the classifier, we have computed the volume of the largest empty N-sphere that can be found inside the one-class classifier. This volume indicates if we can decrease the volume of the one-class classifier, by increasing the complexity parameter of the classifier, without significantly increasing the error on the target class at the same time. The $\mathcal{V}$-statistic has been defined as the ratio of two volumes.

The proposed criterion is feasible for one-class classifiers based on N-spheres and provides an approximation in low dimensional spaces for non-spherical one-class classifiers. It has been shown that the proposed criterion is useful in the prediction of complexity parameters of one-class classifiers as well as discrimination between different classifiers. However, for high dimensional data large amount of objects is needed.

For classifiers like k-centres, k-means we select complexity parameter based on stability criteria in the $\mathcal{V}$-statistic, since such classifiers hardly overfit. For classifiers like `SVDD` and Parzen that both overfit and underfit we select models based on the minimum in the $\mathcal{V}$-statistic.

Finally, we have assumed that the threshold θ of the one-class classifier is given or it is estimated from an application. However, we can also use the presented approach to estimate both the complexity parameter and the threshold. In such case we propose to split the training set in two sets, one to optimise the complexity and the other to optimise the threshold to avoid any bias.

Part III: Accommodation of unlabelled data to enhance classification performance

Find an error, and fix it,
and the classifier will work today.
Show the classifier how to find errors,
and the classifier will work forever.

[4]

[4]modified statement of Oliver G. Selfridge from AI's Greatest Trends and Controversies

Summary of Part III: Accommodation of unlabelled data to enhance classification performance

In the previous chapters we have discussed the problem of one-class classification. This problem arises, for example, when one of the classes, in a classification problem, is well represented and the other classes are too weakly or too broadly defined creating an almost uniformly distributed class of outliers. The inadequate sampling of objects can be one of the reasons to one-class classification problem. This ill-defined problem can rise by the incorrect sampling of objects, i.e. when one samples a classification problem according to the distribution of objects some classes can be undersampled and other oversampled. An other reason for ill-sampling can be the cost of obtaining labelled examples. Examples of some of the classes can be more expensive to measure or label than examples of other classes.

In this part, we investigate problems for which there is a large amount of unlabelled data available but information about class labels is sparse. The problem we are trying to solve is how to improve the performance of classifiers by incorporating additional information from unlabelled data efficiently.

In chapters 7 and 8 active leaning is investigated. In this learning process a learner points out to interesting unlabelled objects and requests their class labels from an expert. Since the expert knowledge is considered to be expensive the learner should minimise the number of requests for labels of unlabelled objects by selecting the most informative objects. The challenge is to select these objects without knowing their class memberships.

In chapter 9 semi-supervised learning is investigated. In this type of learning one incorporates the distribution of unlabelled objects into the training of a classifier. The idea is to improve classification performance by combining information about the distribution of a large set of unlabelled data with class information available from a small training set.

Chapter 7

Active learning

In the traditional approach to statistical learning, one tries to determine a functional dependency between some data inputs and their corresponding outputs, such as their labels. This is usually estimated based on a given, fixed set of labelled examples. Active sampling is an alternative approach to automatic learning: given a pool of unlabelled data X_u, one tries to select a set of training examples in an active way to reach a specified classification error with a minimum cardinality. Therefore, ideally the same classification error is achieved with a smaller number of labelled objects. The criterion of how informative a new object is depends on what we are interested in. We may select a new object according to the following strategies:

1. selected object needs to be maximally informative about values of parameters of an estimated classification model,

2. select objects only from a limited region, e.g. around a region that we are not able to sample directly, to improve classification accuracy only locally

3. select objects such that models that have the largest error are rejected

Before continuing introduce four definitions:

Definition 7.1 *An active learning function $\mathcal{F}$ assigns a real value to each unlabelled object $\mathcal{F}(\mathbf{x}_i) \to \mathbb{R}$, $\mathbf{x}_i \in X_u$. Based on this criterion we can rank unlabelled objects and select the most informative object, $\mathbf{x}^*$, according to $\mathcal{F}$:*

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \mathcal{F}(\mathbf{x}_i) \tag{7.1}$$

The most informative object $\mathbf{x}^* \in X_u$ is the one, that after revealing its label and adding to the training set improves the knowledge about a classification problem the most.

Definition 7.2 *A classifier h with an active learning function $\mathcal{F}$ is called a learner.*

Definition 7.3 *An expert which gives a correct labels for preselected objects is called an oracle.*

Definition 7.4 *A version space is a set of classifiers with a zero error on a training set.*

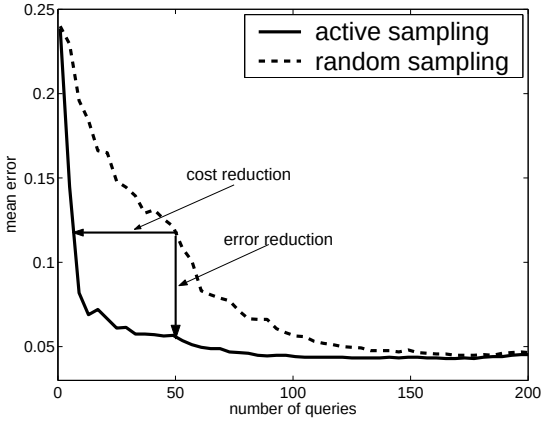


Figure 7.1: Expected learning curves for active and random sampling.

Assume that an initial labelled training set X_t, a classifier h, a active function $\mathcal{F}$ and unlabelled data X_u are given.

1. Train classifier h on the current training set X_t; $h(X_t)$.
2. Select an object $\mathbf{x}^*$ from the unlabelled data X_u according to the active query function $\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \mathcal{F}(\mathbf{x}_i)$.
3. Ask an oracle for the label of $\mathbf{x}^*$. Enlarge the training set X_t and reduce X_u: $X_t = X_t \cup \{\mathbf{x}^*\}$, $X_u = X_u \backslash \{\mathbf{x}^*\}$.
4. Repeat steps (2)–(4) until a stopping criterion is satisfied, e.g. the maximum size of X_t is reached.

Algorithm 7.1: A general framework of pool-based active learning.

Several schemes have been proposed within an active learning framework. In stream-based active learning [Seung et al., 1992, Freund et al., 1997], a classifier is provided with a stream of unlabelled objects. In each trial, a new object is drawn from this stream and presented to the learner, whose task is to decide whether to request its label or not. In the scheme of active learning with membership queries [Angluin, 1988], in each trial, the learner generates an object in an input space and requests its label. There is no pre-defined set of unlabelled objects.

In this section, we focus on pool-based active learning [Lewis and Gale, 1994, Cohn et al., 1995, Tong and Koller, 2000, Roy and McCallum, 2001, Baram et al., 2004, Juszczak and Duin, 2004]. In this learning scheme the learner has access to a small set of labelled objects X_t, a training set, and a large pool of unlabelled objects X_u. Objects are usually selected one by one according to a specified active learning function $\mathcal{F}$. The optimal active learning function selects those objects that, when added to the training set result, for example, in the highest reduction of the classification error. Therefore, in an active learning framework it is assumed that the performance of the current classifier can be improved by enlarging the current training set by additionally selected examples.

Active learning is usually compared to passive learning, in which a learner randomly draws unlabelled objects. The passive learner samples a class distribution with the probability density function $P(X)$. The performance is measured as a generalisation error obtained on an independent test set.

The general framework of pool-based active learning is presented in Algorithm 7.1. Assume that the following are given: an initial training set X_t, a classifier h, an active learning function $\mathcal{F}$, called also a query function, and a pool of unlabelled data X_u. Unlabelled objects are actively selected (usually one by one) from X_u according to the criterion $\mathcal{F}$. After their labels are obtained from an oracle, the training set is enlarged and the classifier h is re-trained. The performance of h is measured on an independent test set. The generalisation error is a

function of an actively selected training set. The resulting learning curve, is usually compared with the learning curve based on random sampling from X_u. A good query function should at least outperform random sampling during at least a part of or the entire learning process, as illustrated in figure 7.1.

To describe specific active learning functions let us introduce first the notation. We are given a C class problem, with labels $\boldsymbol{\omega} = [\omega^{(1)}, \omega^{(2)}, \ldots, \omega^{(c)}]$. h is a classifier, hence a function returning a class assignment for any object $\mathbf{x}_i$. $h(X_t)$ is a classifier h trained on the current set X_t. The decision of h is based on $\omega \equiv \arg\max_{j=[1,\ldots,C]} h(\omega^{(j)}|\mathbf{x}_i)$, where $h(\omega_j|\mathbf{x}_i)$ is the output of h estimating the resemblance of $\mathbf{x}$ to the class $\omega^{(j)}$ (the higher value the higher the resemblance). From the applicability point of view, the class-wise outputs of the classifier are converted to the probability estimates $P(\omega^{(j)}|\mathbf{x}_i)$ by applying a sigmoidal transformation [Tax and Duin, 2002]. Therefore the raw output of a classifier $h(\omega^{(j)}|\mathbf{x}_i)$ is normalised to $P(\omega^{(j)}|\mathbf{x}_i)$ such that $0 \leq P(\omega^{(j)}|\mathbf{x}_i) \leq 1$, $\forall_{j=[1,\ldots,C]}$ and $\sum_{j=1}^{C} P(\omega^{(j)}|\mathbf{x}_i) = 1$.

Several active learning functions are proposed in the literature. Most of them are computed on the normalised output of a classifier, $P(\omega|\mathbf{x})$, and/or directly on the current training set X_t. We denote the active learning function as $\mathcal{F}(\mathbf{x}_i|P(\omega^{(j)}|\mathbf{x}_i), X_t)$. The active learning function $\mathcal{F}$ is computed for all unlabelled objects X_u. An object $\mathbf{x}^* \in X_u$ is selected for which the active learning function takes the maximum value:

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \mathcal{F}(\mathbf{x}_i|P(\omega^{(j)}|\mathbf{x}_i), X_t) \tag{7.2}$$

The set of active learning functions can be divided into four groups according to their utility criterion. The active learning methods select objects according to:

1. an uncertainty measure for label assignments of a single classifier $h(X_t)$ trained on the current training set X_t,

2. the disagreement in a committee of classifiers,

3. the difference in distributions between labelled and unlabelled data sets,

4. the minimisation of the version space.

Note that all the methods, within this chapter, have been proposed in the literature, for two class problems. Here we rewrite the original ideas for more general multi-class problems. **(1)** An example of the first group of sampling methods is the uncertainty sampling (`us`) [Lewis and Gale, 1994]. This method queries unlabelled objects for which the output of the current classifier $P(\omega|\mathbf{x})$, for all classes $\omega^{(j)}$, are the most uncertain:

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \left\{ \sum_{j=1}^{C} 1 - \left| \frac{1}{C} - P(\omega^{(j)}|\mathbf{x}_i) \right| \right\} \tag{7.3}$$

An example of the outputs of uncertainty sampling, for 1-NN is shown in figure 7.2(a) for a simple $2D$ problem. The unlabelled data set X_u is constituted from the systematically distributed grid of the figure.

The other sampling criteria in this group are based, for example, on the entropy or the Gini coefficient of the output of a classifier [Roy and McCallum, 2001]. Since sampling is performed in the vicinity of the current classifier such active sampling functions are suitable

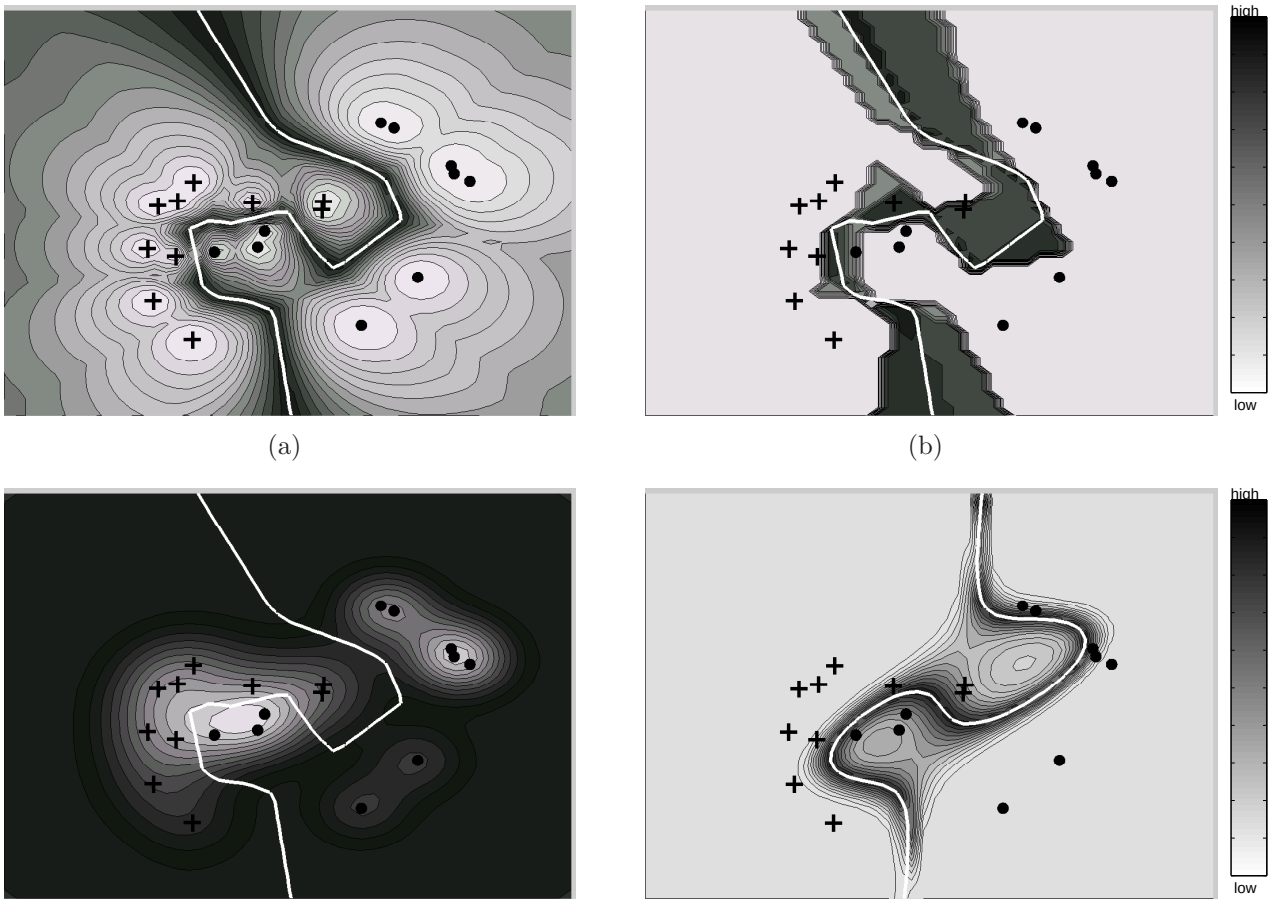


Figure 7.2: The values of active sampling functions: (a) uncertainty sampling (`us`), (b) query-by-bagging (`qbb`), (c) positive density correction (`pdc`) and (d) `MinMax` method, as applied to the figure grids, consisting of systematically distributed unlabelled data X_u. $X_t : \{+, \bullet\}$ indicate the current training set. The current classifier $h(X_t)$ (here the `1-NN` rule in the (a)–(c) plots and the ν-`SVM` for (d)) is represented by a white, solid line. Points with the highest utility scores are coloured in black, while with the lowest utility scores are coloured in white.

for adjusting a classifier in the regions of class overlap. They are not, however, suitable for the search of new classes or new modes of the known classes.

(2) In committee-based approaches [Argamon-Engelson and Dagan, 1999] [Abe and Mamitsuka, 1998, Fine et al., 2002], a committee of classifiers is trained. Each member of the committee labels objects $\mathbf{x}_i \in X_u$. The object for which the committee members disagree the most is chosen as a query to an oracle. [Abe and Mamitsuka, 1998] proposed the query-by-bagging (`qbb`) method, where the committee of classifiers is constructed on J bootstrapped versions X_{t_k}, $k = 1, \ldots, J$, of the training set X_t.

Let $h(X_{t_k})$ denote the k-th classifier trained on X_{t_k}, $X_{t_k} \subset X_t$ and let $\omega^{(k)}$ be a class membership of $\mathbf{x}_i$ estimated by $h(X_{t_k})$, i.e. $\omega^{(k)} = \arg\max_{j=[1,\ldots,C]} P(\omega^{(j)}|\mathbf{x}_i, X_{t_k})$. We assume here that the class labels $\omega^{(k)}$ are numbers $\omega^{(k)} \in [1, \ldots, C]$. The object $\mathbf{x}^* \in X_u$ is selected for which the disagreement about its class membership is the largest over a committee of J

classifiers $h(X_{t_k})$:

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \quad \sum_{k=1}^{J} |\overline{\omega} - \omega^{(k)}| \tag{7.4}$$

$\overline{\omega}$ denotes the mean over the classification labels assigned by committee members $\overline{\omega} = \frac{1}{J} \sum \arg\max_{j=[1,...,C]} P(\omega^{(j)}|\mathbf{x}_i, X_{t_k})$. The illustration of the output of the query-by-bagging function is presented in figure 7.2(b). A single classifier, the `1-NN` rule, trained on the entire X_t is plotted for a comparison. Such a method attempts to minimise the error of a classifier by minimising its variance component caused e.g. by different training sets.

(3) The active sampling methods based on distributions of labelled and unlabelled data sets [Juszczak et al., 2005, Nguyen and Smeulders, 2004] are especially useful for multi-modal or multi-class problems, where more exploratory sampling is needed. [Juszczak et al., 2005], see also figure 7.1(c), proposed an active sampling method, called positive density correction (`pdc`). `pdc` samples unlabelled objects based on the difference in the probability density estimates P as based on the unlabelled objects $P(\mathbf{x}_i|X_u)$ and training objects $P(\mathbf{x}_i|X_t)$:

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \{P(\mathbf{x}_i|X_u) - P(\mathbf{x}_i|X_t)\} \tag{7.5}$$

The method is classifier independent and does not require an initial training set.

(4) The last group of active sampling methods is based on the minimisation of a version space [Mitchell, 1997]. These methods rely either on theoretical works of [Seung et al., 1992, Freund et al., 1997], which also consider committee based sampling, or on recent methods based on the idea of support vector machine (`SVM`). In recent papers [Tong and Koller, 2000, Brinker, 2003, Dasgupta et al., 2005] a version space is approximated by support objects. These methods can only be applied to the `SVM` classifier. An unlabelled object $\mathbf{x}^*$ is selected for which two possible labels split the version space in two equal parts. This means that always half of the classifiers consistent with the current training set are rejected. A tacit assumption is that the distribution of classifiers is uniform i.e. all classifiers are equally probable.

The outputs of the `MinMax` method [Tong and Koller, 2000] is illustrated in figure 7.2(d). Although this sampling is theoretically attractive, it is, in practice, infeasible since it requires a large number, i.e. $C|X_u|$, of different classifiers to be trained to estimate the value of $\mathcal{F}$ for each query [Freund et al., 1997, Tong and Koller, 2000].

In addition, the active learning functions can be divided in three groups according to a state of learning. In the first group, related to explorative learning, a learner 'swaps' a representation space by sampling from regions where it has no labeled examples. `pdc` and `vila` belong to this group, we describe these methods in the next sections. In the second group, related to exploitation learning, a learner samples only within a region between different classes. Consequently, the focus is on a class overlap regions. The last group, relates to random sampling, which allows to learn a probability density function exactly. The difference between exploration and exploitation methods is shown in figure 7.3. The `pdc` and `vila` sample from a remote cluster where there are no labelled objects. Other methods focus on a class overlap region.

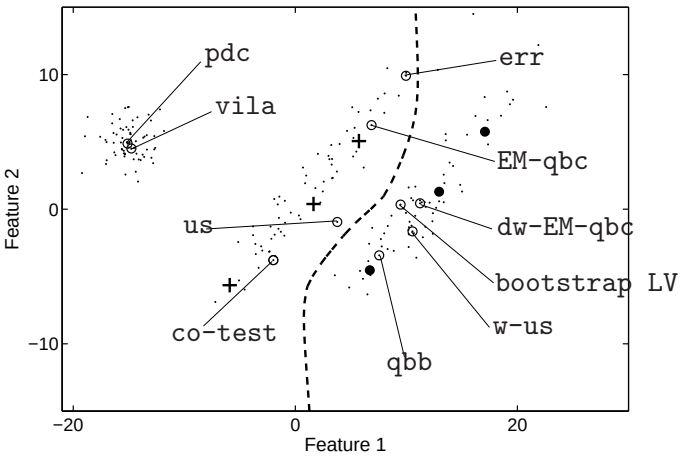


Figure 7.3: Queries selected by several active learning functions. $X_u : \{\cdot\}$, $X_t : \{+, \bullet\}$ and the current classifier is denoted by a dashed line. `err` - estimation of error reduction, `w-us` - weighted `us`, `dw-EM-qbc` - density weighted, EM, query-by-committee, `EM-qbc` EM query-by-committee.

7.1 Active learning based on positive density correction

In most active learning methods presented in the introduction, the classifier plays an important part in the selection of the objects. Decisions are based on some change in classifier parameters, the elimination of a bulk of not-consistent classifiers or on some distance measures to the classifier. These are examples of conservative criteria, where it is assumed that all regions in a feature space where the classification boundary can possibly exist, are known. Therefore they belong to the exploitation group of learners. A second assumption is that a small initial labelled set X_t is given.

However, one can imagine a problem in which there is no access to an initial training set, but there is only a pool of unlabelled objects X_u and even no information about the number of classes is known beforehand. For example, a company would like to learn a response of a group of people, about its new product. The task is to learn how a particular person responds to the new product. Since the product is new no initial training set exists. In addition, the number of examined people should be minimised due to cost.

This section introduces an active learning method which does not require an initial training set. Similar problems are studied in prototype selection [Sánchez et al., 1997] or in selective sampling [Angluin, 1988]. However, in our case objects are selectively drawn from the pool of unlabelled objects and they are not generated from distributions to be learned. The proposed selection criterion is based on the difference in the density estimates for the training set X_t $P(\mathbf{x}_i|X_t)$, and the unlabelled set X_u $P(\mathbf{x}_i|X_u)$:

$$\mathbf{x}^* \equiv \arg\max_{\mathbf{x}_i \in X_u} \left\{ P(\mathbf{x}_i|X_u) - P(\mathbf{x}_i|X_t) \right\}. \tag{7.6}$$

where P is estimated preferably by a nonparametric, asymptotically conditional density estimation like the `Parzen` estimator. Note that there is no absolute value of the density difference

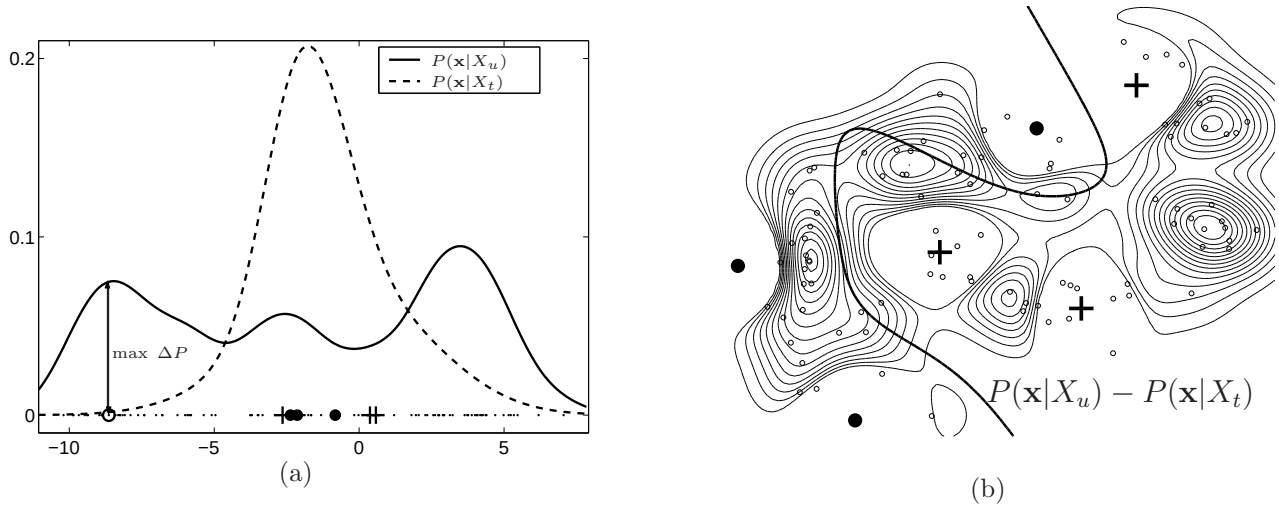


Figure 7.4: (a) A single query $\{\odot\}$ selected according to the positive density correction method `pdc`. $\{\bullet, +\}$ training set X_t and $\{\cdot\}$ unlabelled set X_u. (b) The density difference between $P(\mathbf{x}|X_u)$ and $P(\mathbf{x}|X_t)$ are indicated by isolines.

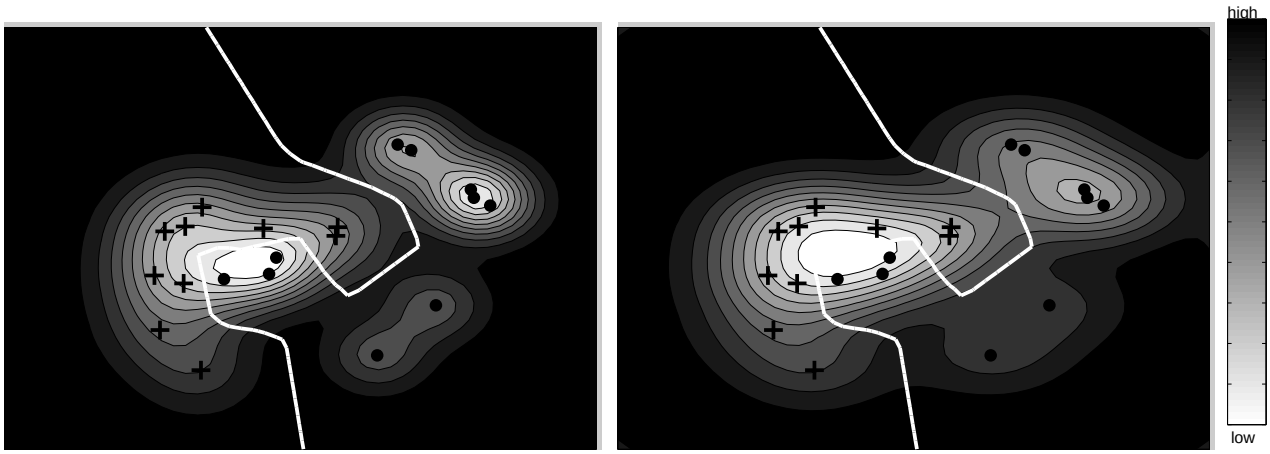
taken. Consequently, we add objects from the region where $P(\mathbf{x}_i|X_t)$ is low. Therefore, it is called the positive density correction (`pdc`). A single query selection is illustrated in figure 7.4 and in figure 7.5 the score of the sampling criterion based on the difference in density estimates for a toy problem is shown. The `pdc` algorithm is summarised in Algorithm 7.2.

1. Assume only an unlabelled set X_u is given.
2. Select, from X_u, the object, $\mathbf{x}^*$, with the highest estimated density $P(\mathbf{x}_i|X_u)$ to be labelled by an expert and add it to X_t, $X_t := \{\mathbf{x}^*, \omega\}$, $X_u := X_u \backslash \{\mathbf{x}^*\}$.
3. Compute the difference in density estimates between the labelled set X_t and the unlabelled set X_u: $P(\mathbf{x}_i|X_u) - P(\mathbf{x}_i|X_t)$, $\forall_{\mathbf{x}_i \in X_u}$.
4. Select an object $\mathbf{x}^* \in X_u$ to be labelled by an expert with the maximum value for the density difference and add it to X_t, $X_t := X_t \cup \{\mathbf{x}^*, \omega\}$, $X_u := X_u \backslash \{\mathbf{x}^*\}$.
5. Repeat $3-4$ until the required stopping criterion is reached e.g. the size of the training set is maximum.

Algorithm 7.2: Active learning based on the positive density correction.

Since the `pdc` method can be used without an initial training set X_t, in contrast to other active sampling methods, we compare it in this section only with random sampling. In the next section `pdc` is compared with other active learning functions using an initial training set.

In figure 7.6 some results of experiments on the UCI repository datasets are presented. Datasets were split into two parts: the unlabelled set X_u and the test set. The size of the unlabelled set is indicated by the length of the abscissa. To compute densities on X_t and X_u in equation (7.6) the `Parzen` density estimator [Parzen, 1962] was used, with the smoothing parameter optimised by the maximum likelihood criterion [Duin, 1976]. In experiments presented in figure 7.6 the density of X_t was estimated per class to use all available information.



(a) (b)

Figure 7.5: The output for X_u constituting of the grid of a figure, computed by the sampling method based on the positive density correction for the labelled set X_t and the unlabelled set X_u. In (a) the density of X_t is estimated per class (b) the density of X_t is estimated without information about classes. The white, continuous line denotes the current classifier.

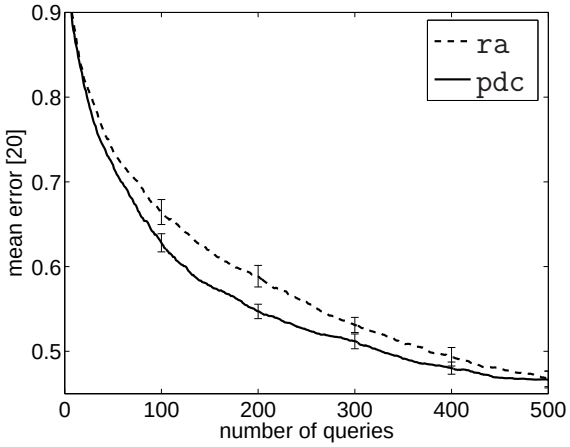
Moreover, by estimating density per class the class unbalance is avoided during sampling.

In step 3 of Algorithm 7.2, when $|X_t| = 1$ or when there is only a single object for one of the classes in X_t, the smoothing parameter for the `Parzen` density estimation can not be determined by the maximum likelihood criterion. For this special situation the smoothing parameter was determined by the distance of the single object $\mathbf{x} \in X_t$ to its nearest neighbour in X_u.

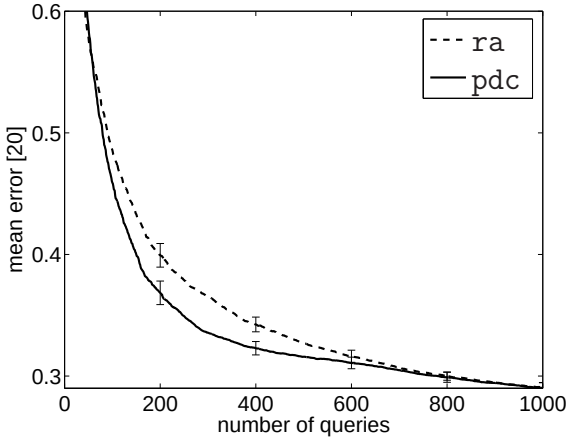
The performance of active learning method was estimated using the one-nearest-neighbour rule because none of the other classifiers can be computed and optimised over the entire learning process. In most learning problems that were examined, learning curves look like these of *chromo*, *cbands* and *diabetes* datasets, presented in figure 7.6. The selective sampling `pdc` outperformed random sampling especially when datasets have a high number of modes (*diabetes*) or classes (*chromo*, *cbands*). In figures 7.6(d) and 7.6(f), datasets projected to first three `PCA` components are presented. One can see that *diabetes* dataset has several modes distributed along a cigar shaped structure. Clearly, if queries are chosen according to the proposed method `pdc`, modes of the dataset can be explored faster than when queries are selected randomly, which can be seen in the learning curve on the corresponding left figure.

For the *heart* dataset it is possible to select, using the `pdc` method, a subset of objects for which the one-nearest neighbour error is lower than when the entire set X_u is included in the training set. This result is due to the classifier that has been chosen. The complexity of the nearest neighbour classifier is related to the number of prototypes, the size of X_t. From the learning curve it can be seen that after adding about 25 objects the one-nearest neighbour classifier becomes overtrained. To avoid overtraining when the size of X_t increases k, the number of nearest neighbours, should also increase. In the presented experiments $k = 1$ over the entire learning curve to have a consistent error measure. This result indicates that `pdc` can be used also as a prototype selection method.

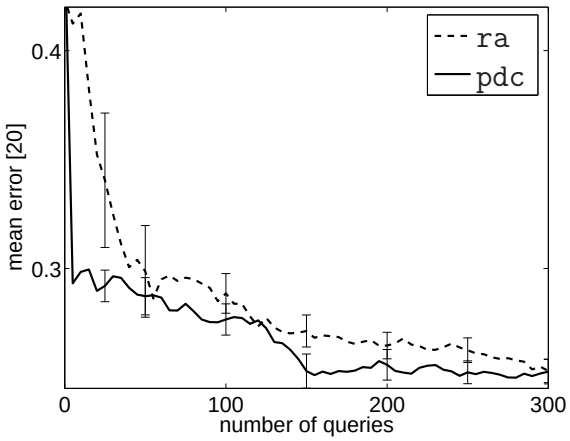
In the experiments on the *wbc* dataset (*Wisconsin breast cancer* dataset) random sampling has similar performance to the `pdc` method, figure 7.7(b). Although, the differences are not significant, the phenomenon is interesting to mention here. From the 3D `PCA` projection it



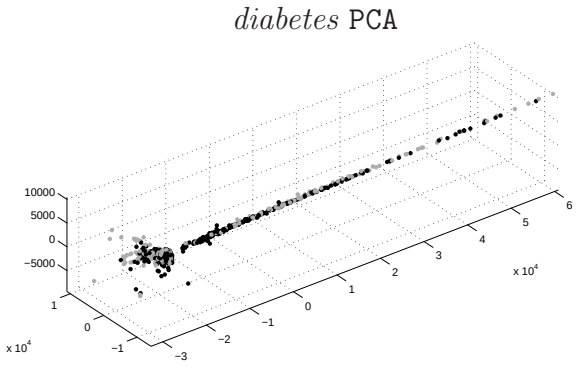
(a) *chromo*



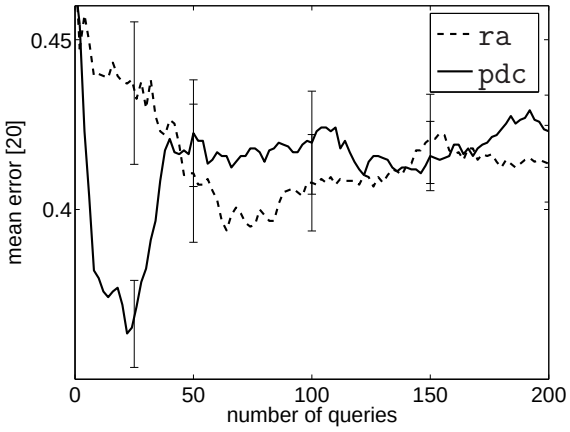
(b) *cbands*



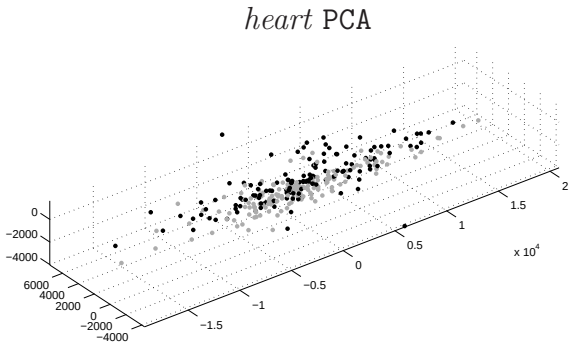
(c) *diabetes*



(d)



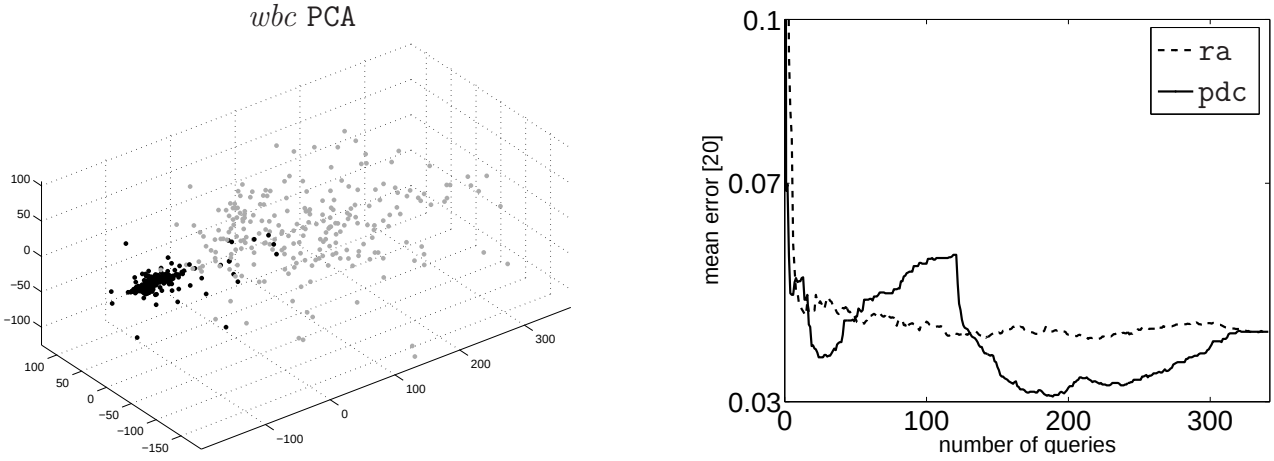
(e) *heart*



(f)

Figure 7.6: Learning curves for random sampling `ra` and `pdc` sampling for some of the UCI repository datasets, figures (a,b,c,e), averaged over 20 trials. In figures (d,f) the $3D$ `PCA` projections of *diabetes* and *heart* datasets are shown.

can be seen that, although the classification problem is simple, one of the classes is highly compact compared to the other one. This means that in the beginning of the learning process



(a) (b)

Figure 7.7: (a) 3D PCA projection and learning curves for the *Wisconsin breast cancer* dataset. (b) The learning curve for the 1-NN rule trained on objects selected by random and positive density correction methods. averaged over 20 trials.

more examples are sampled from the more dense class. Then examples from the second class are sampled, and so on. Therefore the learning curve for pdc oscillates around the learning curve for random sampling.

7.2 Active learning based on the variation in label assignments

The active learning methods presented in the literature perform well when the current classifier is in the vicinity of the optimal classifier. Methods like the uncertainty sampling, query-by-bagging and the estimation-of-error-reduction often fail on multi-modal datasets by selecting objects that refine the current model instead of investigating the feature space; see figure 7.3. MinMax sampling works well but only if the parameters of SVM are correctly estimated. In general, the presented methods use variation in the classifier parameters or a distance measure to a decision function as a measure of utility of an unlabelled object. This is not necessarily related to a gain in a classification performance.

In this section, a new active learning method is presented.

The method is based on the assumption that the learning problem is difficult, e.g. multi-modal, and the current classifier is far from the optimal solution. In such problems, the current learner has to explore the feature space. The proposed technique relies on the variation in label assignments for the unlabelled dataset X_u. The proposed active learning function for a single object $\mathbf{x}_i \in X_u$ is computed on the variation in label assignments for $X_{u_{-i}} := X_u \backslash \{\mathbf{x}_i\}$ between a classifier $h(X_t)$ trained on a current training set X_t and the set of C classifiers trained on the enlarged training set X_t with all possible labels for $\mathbf{x}_i$, $X_t \cup \{\mathbf{x}_i, \omega^{(j)}\}$. If we define ω_j^{u-i} as a set of labels of $X_{u_{-i}}$ assigned by the classifier $h(X_t \cup \{\mathbf{x}_i, \omega^{(j)}\})$ and $\omega_{u_{-i}}$ as a

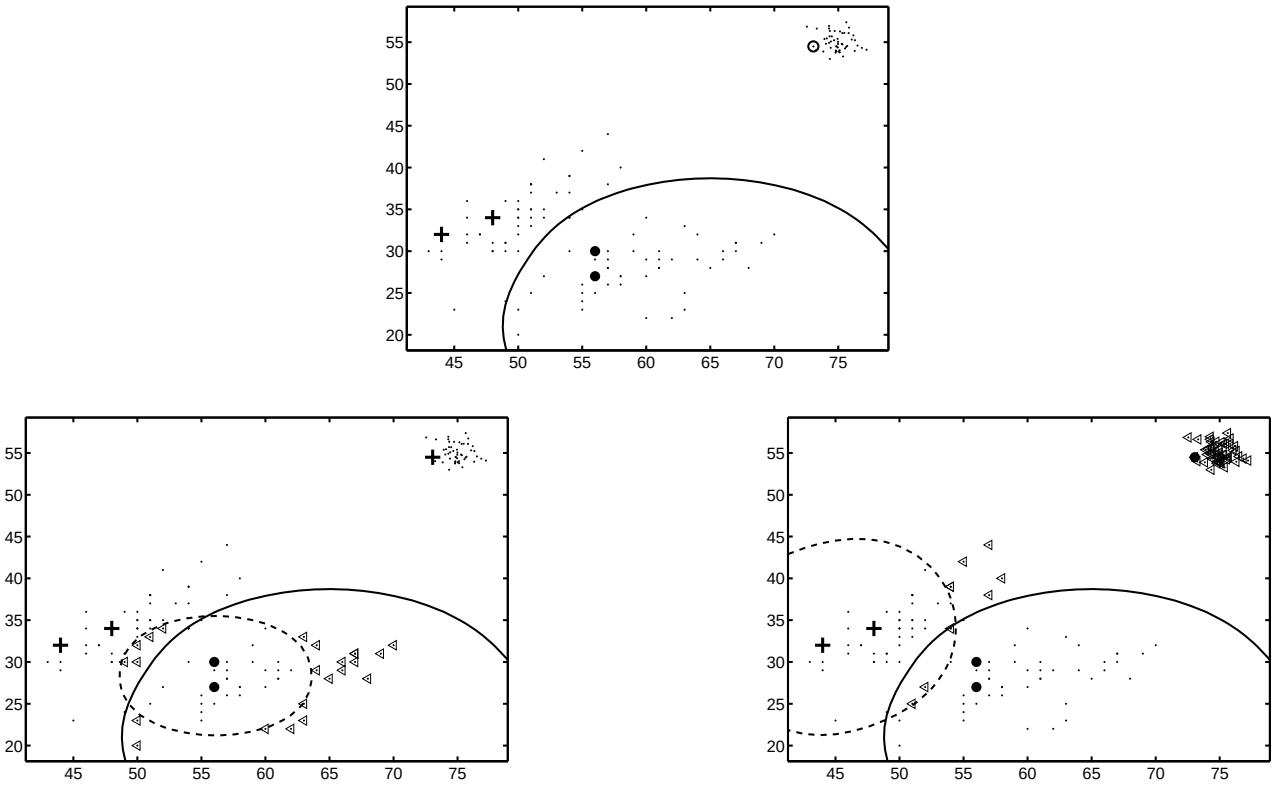
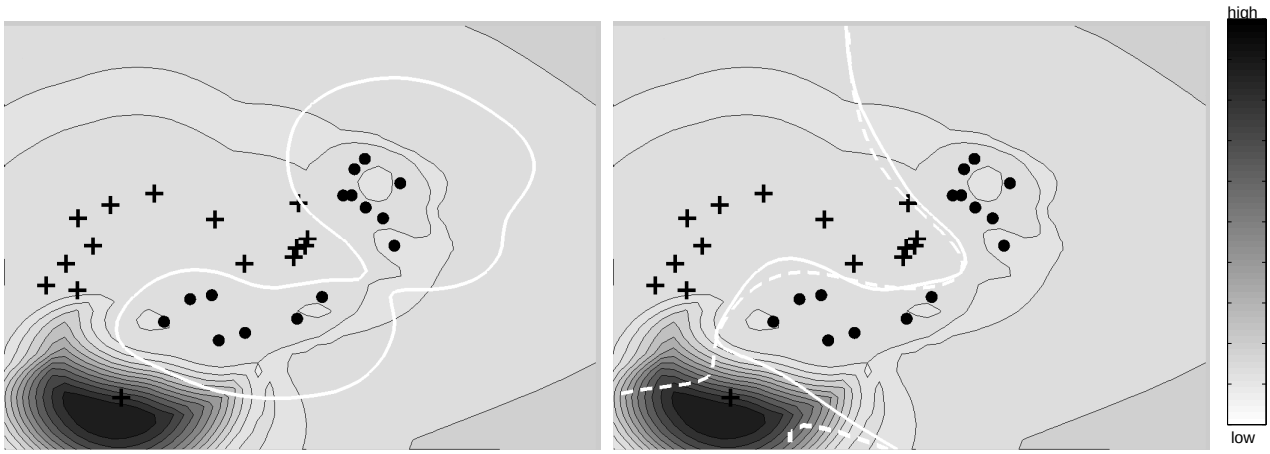


Figure 7.8: A single query selection based on the variation in labels assignments (`vila`). The potential query is marked by $\{\odot\}$ and the solid line denotes the current classifier, here `Parzen`. In the left figure the query is assigned to the $\{+\}$ class and in the right figure to the $\{\bullet\}$ class. The corresponding classifiers are drawn as the dashed line. Objects that change labels are marked by $\{\triangleleft\}$.

set of labels assigned by the classifier $h(X_t)$, then the proposed criterion can be written as:

$$\mathbf{x}^* = \max_{\mathbf{x}_i \in X_u} \min_{j \in [1,\ldots,C]} \sum_{X_{u_{-i}}} \left[\mathcal{I}\{\omega^{u_{-i}}_{[\mathbf{x}_k]} \neq \omega^{u_{-i}(j)}_{[\mathbf{x}_k]}\} \right] \tag{7.7}$$

The error of the classifier is expected to be reduced by selecting those objects which cause the largest change in the label assignments of an unlabelled set X_u for all labels $\omega^{(j)}$. To avoid selection of outlier objects, the label which gives minimum number of changes in label assignments is computed in equation (7.7). This means that at least this number of labels



(a) (b)

Figure 7.9: (a) The output for the grid of a figure constituting X_u, computed by the sampling method based on the variation in label assignments. The white, solid line denotes the current classifier, here `Parzen`. (b) Two classifiers, dashed and solid lines, obtained after adding the object from the centre of the region of the highest score for the two possible labels.

change after revealing the true label of $\mathbf{x}^*$.

1. Let an initial training set X_t and an unlabelled set X_u be given.
2. Train the classifier $h(X_t)$ on the labelled set X_t.
3. Let $X_{u_{-i}} := X_u \backslash \{\mathbf{x}_i\}$. Classify $X_{u_{-i}}$ by $h(X_t)$, $\omega^{u-i} = \arg\max_j P(\omega^{(j)}|X_{u_{-i}}, h(X_t))$
4. Add $\mathbf{x}_i$ to the temporary training set with the assumed label $\omega^{(j)}$, $X_t \cup \{\mathbf{x}_i, \omega^{(j)}\}$.
5. Train the classifier $h(X_t \cup \{\mathbf{x}_i, \omega^{(j)}\})$ and classify $X_{u_{-i}}$, $\omega^{u-i(j)} = \arg\max_j P(\omega^{(j)}|X_{u_{-i}}, h(X_t \cup \{\mathbf{x}_i, \omega^{(j)}\}))$.
6. Repeat the steps $4-5$ for all labels $\boldsymbol{\omega} = [\omega^{(1)}, \ldots, \omega^{(C)}]$.
7. Repeat $3-6$ for all $\mathbf{x}_i \in X_u$.
8. Select $\mathbf{x}^*$ that maximise equation (7.7) to be labelled by an expert and add it to X_t, $X_t := X_t \cup \{\mathbf{x}^*, \omega\}$, $X_u := X_u \backslash \{\mathbf{x}^*\}$.
9. Stop if a stopping criterion is fulfilled, e.g. when the training set has the maximum size.

Algorithm 7.3: The variation in label assignments (`vila`).

An example of a single query selection is shown on figure 7.8 and the general algorithm is presented in Algorithm 7.3. In figure 7.9(a) the calculated output for our toy problem by the proposed method based on the change in label assignments is presented. In figure 7.9(b) two classifiers represented by dashed and solid lines after including the object from the figure grid, with the highest score (the centre of the black region in figure 7.9), for the two possible labels are shown.

We have compared the performance of both proposed sampling methods `pdc` and `vila` in

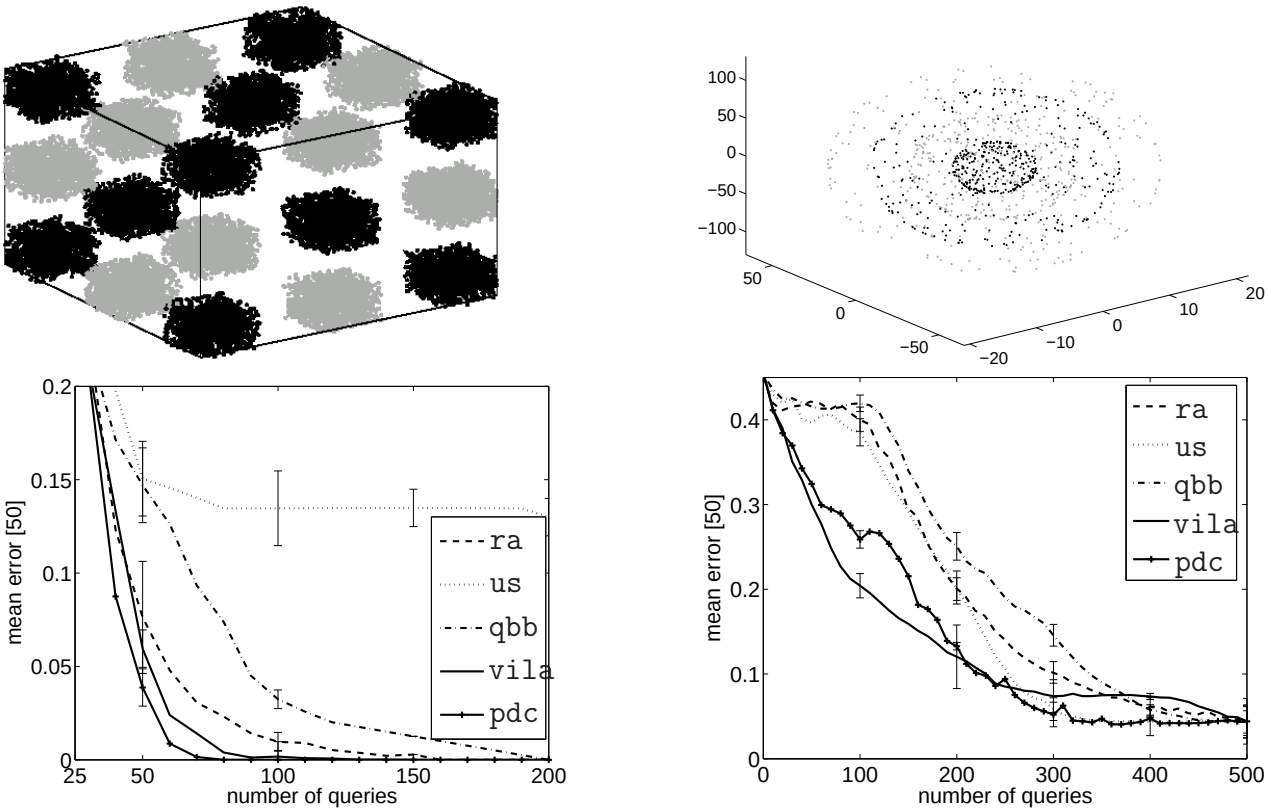
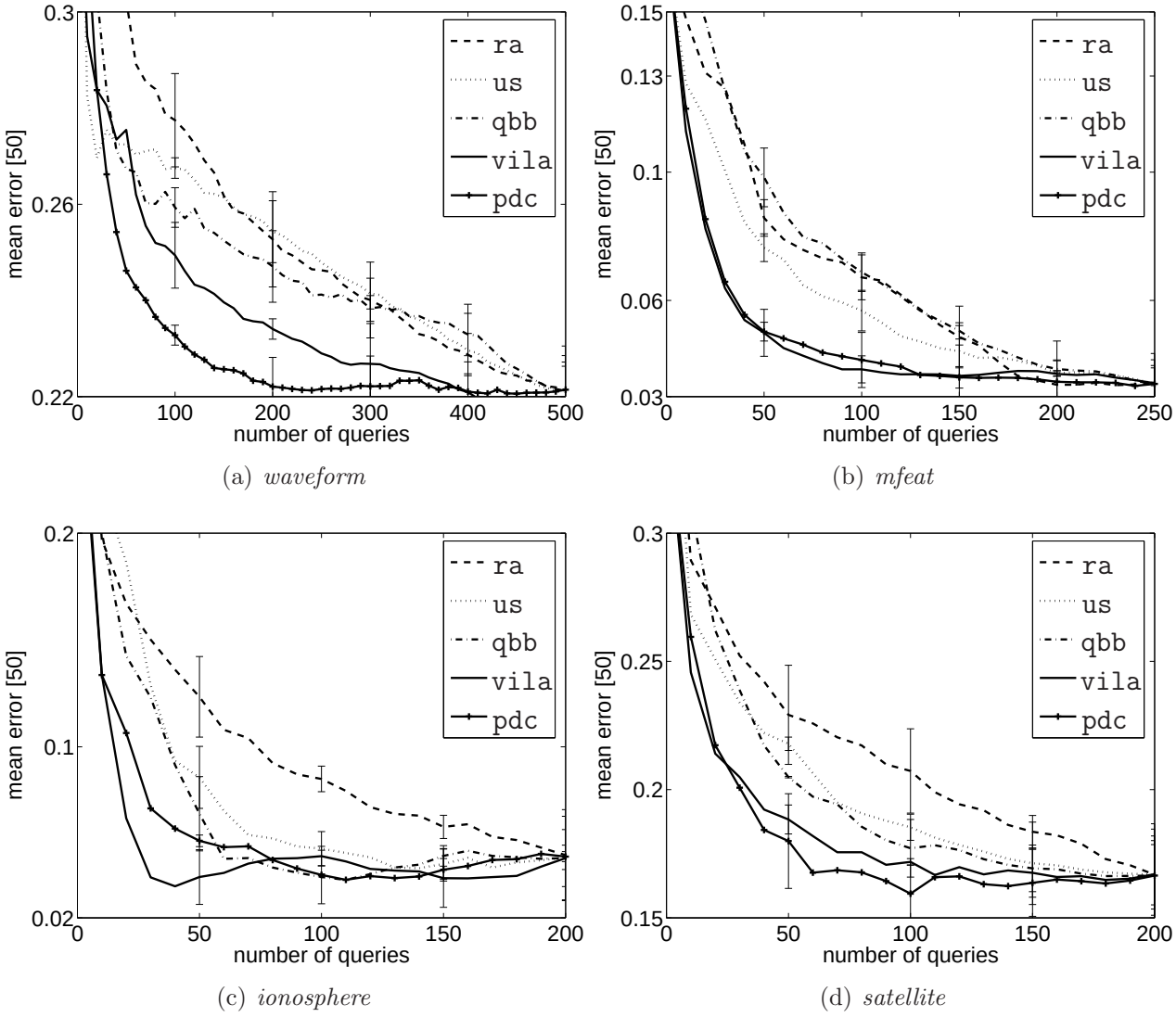


Figure 7.10: Learning curves for the $3D$, non-overlapping, multi-modal two class problems. `ra` random sampling, `us` uncertainty sampling, `qbb` query-by-bagging, `vila` variation in label assignments, `pdc` positive density correction.

situations where an initial training set is given. The performance of the proposed query functions are also compared to the random, uncertainty and query-by-bagging sampling methods on some toy problems and real-world datasets. These sampling methods were selected because they are de facto standards in the field of the active learning. Moreover, uncertainty sampling is an example of conservative sampling and both these active sampling methods heavily rely on the classifier.

All experiments were performed using the `Parzen` classifier. Datasets were split in three parts: the initial training set X_t consisting of two objects per class, the initial unlabeled training set X_u and an independent test set.

In figure 7.10, learning curves for $3D$, non-overlapping, multi-modal, two class problems are presented. The size of X_u equals 500 objects. It can be seen that `pdc` and `vila` outperform the other sampling criteria. For the separated $3D$ *chess board* dataset, the proposed active learning methods required $60 - 80$ queries to reach zero error on the test set, compared to random and `qbb` requiring about 200 queries. Uncertainty sampling requires about 300 objects to reach the same error. It can be clearly seen that sampling close to the current decision boundary according to uncertainty sampling and query-by-committee the performance improvement is worse than by using random sampling. The same can be observed in the figure on the right for a bit more difficult problem of enclosed 3D ellipsoids. The maximum error difference, in this case between: `us`, `qbb`, `ra` and the proposed methods: `pdc` and `vila` is about 20%, on the independent test set, for the first 100 queries.



(a) *waveform* (b) *mfeat*

(c) *ionosphere* (d) *satellite*

Figure 7.11: Learning curves for UCI datasets (average over 50 trials).

Figure 7.11 shows examples of learning curves for several UCI datasets [Hettich et al., 1998]. The setting of experiments is the same as in the toy example. It can be observed that `vila` and `pdc` perform better, compared to other sampling techniques, when a dataset has more classes. This can be related to more complicated structure in the dataset itself, e.g. many modes.

Presented methods belong to the exploration group of learners, where it is assumed that a problem is complicated, e.g. multi-modal, is such problems they provide faster convergence of the classifier to a small error.

7.3 On the choice of a classifier in the active learning framework

In the previous sections we have discussed active learning techniques that can be applied with any type of classifier, like `vila`, or that do not use a classifier at all, like the `pdc`. However, some active learning methods proposed in the literature use specific features of the chosen classifier in the definition of the utility function. These active learning methods have been designed for particular classifiers: support vector machine [Tong and Koller, 2000], self-organising maps [Hasenjäger et al., 1999], k-nearest neighbour classifier [Lindenbaum et al., 2004], naïve Bayes classifier [Roy and McCallum, 2001] or Parzen classifier [Chapelle, 2004]. All of these learning techniques can be used only with the particular type of classifier, they were designed for. However, the question remains which classifier, or type of classifiers (e.g. distance based, density based), should be chosen for an active learning procedure? In this section some general features of a particular classifier in relation to active learning are described.

The performance of a learner depends on the classifier, the sampling method and the problem. In general, no classifier can be preferred over any other one if there is no prior knowledge available about the classification problem; `the no-free lunch theorem`, [Wolpert and Macready, 1997]. However, because in general some of the difficulties exist in the active learning framework, e.g. the small sample size problem in the beginning of learning, unbalanced training data, a set of desired properties of a classifier could be described. In general a classifier that is used in the active learning framework should:

1. have the possibility of adapting its complexity to that of the problem, from linear in the beginning of the learning process to possibly highly nonlinear decision boundaries, to avoid overtraining at the start and to make use of information obtained from new labelled objects.

2. have few parameters to estimate

3. work well in small sample size problems

4. make use of every new labelled object as labeling is expensive

In the last few years, the tendency is to use `SVM` in the active learning framework [Tong and Koller, 2000, Schohn and Cohn, 2000, Baram et al., 2004]. This is quite understandable, because it is flexible, simple and works well in small sample size problems and unbalanced data. However, it has also some drawbacks: selection of the kernel and the trade-off parameter C is still an active area of research. For a survey, see [Seeger, 2004]. Moreover, not all information from every labelled example is incorporated as a change in the classifier itself, and as a result the utility function based on the classifier, is not changed. One can imagine that for limited resources for labeling new objects it is required that every new object should improve a sampling criterion. Therefore, for the `SVM` only unlabelled examples that have a chance to become support objects should be queried. Because of the high labeling cost the exploration possibility of the utility function, in this case, is limited. Consequently, methods that predict change in the classifier should be used e.g. the `MaxMin Margin`, `Ratio Margin` proposed in [Tong and Koller, 2000] or `vila` as proposed in the last section. However, when sampling methods like uncertainty sampling, estimation-of-error-reduction or `pdc` are used for the `SVM` this results in step-like learning curve. This may result in that, despite the

increase in the training set, size the classifier and the utility function stay the same, thus an identical (or similar) query is selected next.

Following the above discussion and because we would like to have a simple and at the same time flexible classifier, with parameters that should be learned from labelled objects and as the classifier should utilise information from every labelled object, in the experiments presented Parzen classifier [Parzen, 1962] was chosen, as a more suitable classifier.

7.4 Conclusions

In this chapter, we have proposed two active learning functions. In comparison to some existing active learning functions the proposed ones have more exploratory properties. Therefore, they outperform most of existing active learning functions in difficult problems, e.g. in multi-modal problems. On the other hand, in problems where classes heavily overlap the exploratory active learning functions might outperform the proposed ones.

In addition, we have discussed the link between classifiers and active learning functions pointing to advantages and disadvantages. It is important, for a performance of a learner (classifier + active learning function), to select a type of a classifier and an active learning function together, taking into account they characteristics. If the classifier is based on global statistic, e.g. a mean or covariance, the active learning function should select objects which give the best estimates of these statistics. On the other hand, if the classifier is only based on few prototypes, from the training set, the active learning function should choose the best prototypes.

Chapter 8

Query diversification in active leaning

In the previous section, examples of active learning methods that select a single query $\mathbf{x}^*$ from a pool of unlabelled objects X_u have been presented. The goal has been to minimise the size of a training set necessary to attain a certain classification accuracy. This is achieved by retraining a classifier each time a label was retrieved for a single query. The more complex task of selecting multiple unlabelled objects as a batch is addressed in this section. In practical problems, there is a need to select a batch of unlabelled objects instead of a single one.

Imagine a fully automatic recognition system that we would evaluate once per month and adjust it if needed. However, during a single month, such a system performs many recognition tasks and stores much more new data than a human expert can verify in the next couple of months. Therefore, it is necessary to pre-select a subset of unlabelled objects that, e.g. have a large probability of misclassification. As they are boundary cases, they should be included to re-train the classifier.

Similarly to the active learning approach, where one draws a single unlabelled object, here the goal is to learn an input-output mapping $X \rightarrow \boldsymbol{\omega}$ from a set of n training examples $X_t = \{\mathbf{x}_i, \omega_i\}_{i=1}^n$, where $\mathbf{x}_i \in X$, $\omega_i^{(j)} \in [\omega^{(1)}, \ldots, \omega^{(C)}]$ and the size n of the training set should be minimised. However now, at each iteration, the active learner is allowed to select a multiple, new training input $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^* \in X_u$ of k elements from the unlabelled data X_u. Note that k can be much larger than 1. The selection of $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^*$ may be viewed as a query based on the following active sampling function:

$$\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^* \equiv \arg \max_{\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^* \in X_u} \sum_{i=1}^{k} \mathcal{F}(\mathbf{x}_i | h(X_t), X_t). \tag{8.1}$$

Having selected $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^*$, the learner is provided with the corresponding labels $[\omega_1, \ldots, \omega_k]$ by an expert, as a result of an experiment or by some other action. The new training input $\{\mathbf{x}_j^*, \omega_j\}_{j=1}^k$ is added to the current training set $X_t = X_t \cup \{\mathbf{x}_j^*, \omega_j\}_{j=1}^k$ and removed from the unlabelled set $X_u = X_u \backslash \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^*$. The classifier is retrained and the learner selects another set $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}^*$. The process is repeated until the resources are exhausted, the training set reached a maximum number of examples, or the error stabilises.

In figure 7.2, the value of different selective sampling functions $\mathcal{F}$ is computed for the systematically distributed unlabelled data X_u, i.e. the points in the figure grid. As it can be observed, the values of different active sampling criteria differ, however they are smooth, they give the same or similar values for neighbouring objects in each sampling method. Therefore, for a large X_u, objects that are in a close neighbourhood, hence likely to contain similar information about the classification problem, are selected in a single draw by a single active

learning function. We argue that, in general, adding a batch of new objects to the training set, selected based on such a criterion, does not necessarily yield a larger improvement of the classification accuracy, than by simply adding one of them. Objects close to each other give similar information.

For an illustration, we first examine a linearly separable problem, as shown in figure 8.1. We know from PAC learning [Valiant, 1984, Haussler, 1990] that for a passive learner, which draws objects randomly, the number of training objects n needed to learn a classification problem with a classification error ϵ and the probability δ is $\mathcal{O}(\ln|H|, \frac{1}{\epsilon}, \ln\frac{1}{\delta})$. $|H|$ denotes the VC-dimension or a countable set of classifiers consistent with the labels of the training set and δ is the probability of failure. On the other hand, if we would like to have a classifier with a zero classification error, $\epsilon = 0$, as e.g. for the problem in figure 8.1, then, in the worst case, we have to examine all $m = |X_u|$ unlabelled objects.

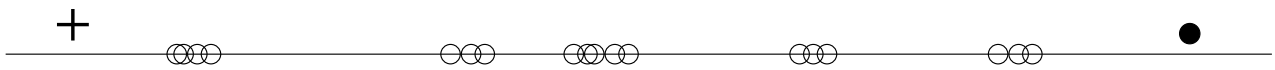


Figure 8.1: Linearly separable, one dimensional problem. $X_t : \{+, \bullet\},\ X_u : \{\circ\}$

However, a binary search requires here only $\log|X_u|$ samples, which gives an exponential improvement over the passive learning [Freund et al., 1997]. Finally, if we explore the distribution of the unlabelled data $\{\circ\}$, positioned in $k = 5$ clusters, it is required to label just k unlabelled objects to reach the zero error classifier, assuming that objects in a cluster belong to the same class. This simple example demonstrates the importance of inspecting the distribution of unlabelled objects, when minimising the number of objects, necessary to achieve a certain classification error.

Since in the standard active sampling algorithms, the classifier is recomputed after each query, the values of an active sampling function $\mathcal{F}$, based on such a classifier, change as well; see Algorithm 7.1. Therefore, unlabelled objects, that are queried, differ in consecutive draws. However, if we consider the simultaneous selection of multiple objects, $k > 1$, similar objects are selected in a single draw. In figure 8.2(a) five objects with the highest values of two active learning functions, `pdc` and `MinMax`, are marked. It can be observed that for each sampling method the selected objects are chosen from the same region in the feature space. These active learning methods do not consider the influence of revealing a label of a single query on the remaining queries. Therefore, adding such a batch is not significantly more beneficial than adding just its single representative. For clarity of a figure we only show these two methods, however the same problem holds for all active learning methods that select a single query.

When we would like to select multiple queries in active sampling we should consider not only the criterion they are based on, e.g. the uncertainty of labels of unlabelled objects, but also the effect of obtaining a class label of a single candidate object on the remaining candidate objects. A straightforward approach to deal with this problem is to select a batch of unlabelled objects that have high values of an active learning function and are maximally informative, i.e. they have a minimum influence on the potential classification labels of other selected objects and high information about not yet selected unlabelled objects. One of the possible approaches might be to select centres of k clusters of X_u. However, such a strategy has a severe drawback as selecting the cluster centres for a large unlabelled set result in the selection of similar objects in consecutive draws. Such consecutive batches of unlabelled objects are similar as removing a few objects from X_u, does not change the estimated cluster centres. The next section discusses an alternative.

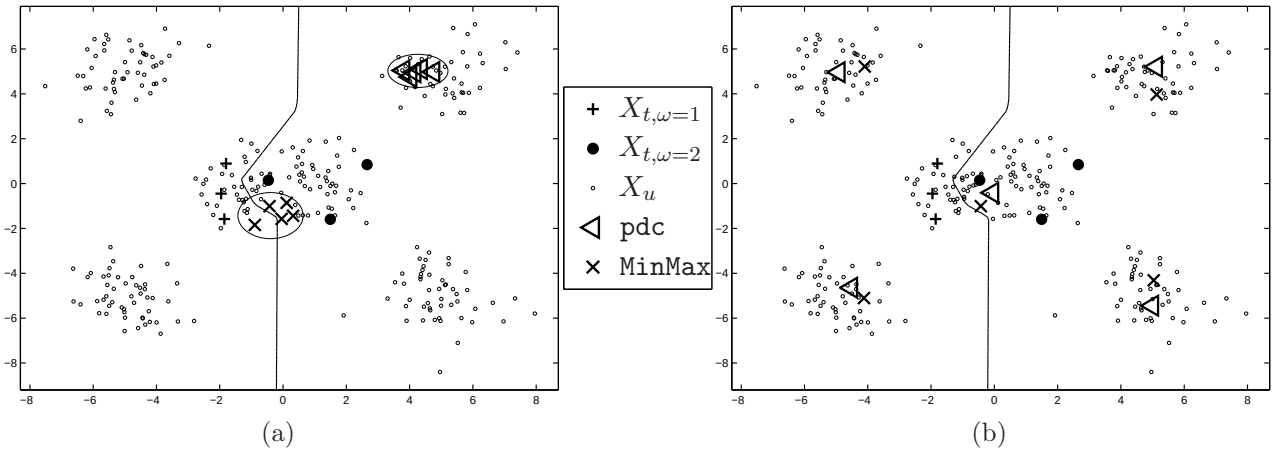


Figure 8.2: (a) Five objects selected by `MinMax` and `pdc` active learning functions. The current classifier, `1-NN`, is drawn as solid line. (b) Five objects selected by `MinMax` and `pdc` active learning functions considering also the distribution of X_u.

8.1 Query diversification based on quadratic programming

We formulate three active query selection algorithms suitable for the selection of informative batches of unlabelled data. The diversification criterion in these algorithms is related to the type of the classifier, e.g. distance or density based, that is going to be trained on the selected objects. The main argument to do so, apart from experimental results, is that such 'personalisation' of queries is helpful in human learning. Different people may require different examples to learn efficiently certain concepts, especially difficult ones. The same also holds for classifiers. For different classifiers selecting unlabelled objects $\mathbf{x}^*$ e.g. with a maximum uncertainty changes the decision boundary locally, e.g. for the `1-NN` rule. Selecting the same object for a parametric classifier, e.g. the mixture of Gaussians classifier changes the decision boundary globally. This also holds for the `SVM`. Therefore, a good sampling function should also include, in its estimation of the utility of a potential query, properties of the classifier itself.

In particular, we propose three active sampling methods with diversification criteria based on distances, densities and inner products between labelled and unlabelled objects. The presented three active sampling methods are generic and can be used with any type of a classifier, however, because they compute the utility criteria in a certain way they are especially useful for classifiers that are based on the same principles e.g. the `1-NN` rule, the Parzen classifier and the `SVM`.

8.1.1 Distance-based diversification

As was mentioned in the last section the most informative batch of unlabelled objects should contain objects that have the minimum influence on the classification labels of each other. Intuitively, this can be related to distances between objects in a batch i.e. by maximising the sum of distances $\sum_i^k \sum_j^k D(\mathbf{x}_i, \mathbf{x}_j)$ between objects to be selected. For small distances we expect redundant class information. If we give a weight $0 \leq \alpha_i \leq 1$ to each object

$\mathbf{x}_i \in X_u$, the above sum can be written as $\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T D\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^T\mathbf{1} = k$ for a sparse solution. Additionally, we are interested in the objects that have information about labels of other not yet selected unlabelled objects, e.g centres of clusters in X_u can be expected to describe remaining unlabelled data in these clusters. To impose this, we can simply demand that the distance $D_{nn}(\mathbf{x}_i) = \|\mu_{nn}(\mathbf{x}_i) - \mathbf{x}_i\|$ between object $\mathbf{x}_i$ and the mean $\mu_{nn}(\mathbf{x}_i)$ of its nearest neighbours should be minimum; see figure 8.3. Note that $\mu_{nn}(\mathbf{x}_i)$ is computed on the set of nearest neighbours of $\mathbf{x}_i$ but without $\mathbf{x}_i$. Since $D_{nn}(\mathbf{x}_i)$ describes only a single object this linear term can be subtracted from the previous quadratic term as $\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T D\boldsymbol{\alpha} - \boldsymbol{\alpha}\mathbf{D}_{nn}$, $\boldsymbol{\alpha}^T\mathbf{1} = k$. Moreover, an active learning function $\mathcal{F}(\mathbf{x})$ should have a high value for the selected objects. This is also a linear term, therefore the selection of objects with the highest values for these three criteria can be written as $\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T D\boldsymbol{\alpha} + \boldsymbol{\alpha}^T(\mathcal{F} - \mathbf{D}_{nn})$, $\boldsymbol{\alpha}^T\mathbf{1} = k$. We compute the utility of the batch of k unlabelled objects by maximising the above formula using a quadratic programming technique. This allows us to optimise the entire batch at once compared to iterative procedures which examine a single object in the batch at a time, as described in the next section.

The diversification of queries, for a particular active learning function $\mathcal{F}$, based on distances is written as:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \boldsymbol{\alpha}^T\mathcal{D}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\boldsymbol{\rho}, \\ \text{s.t.} \quad & \boldsymbol{\alpha}^T\mathbf{1} = k; \quad 0 \le \alpha_i \le 1, \\ & \boldsymbol{\rho} = \mathcal{F} - \mathbf{D}_{nn}. \end{aligned} \tag{8.2}$$

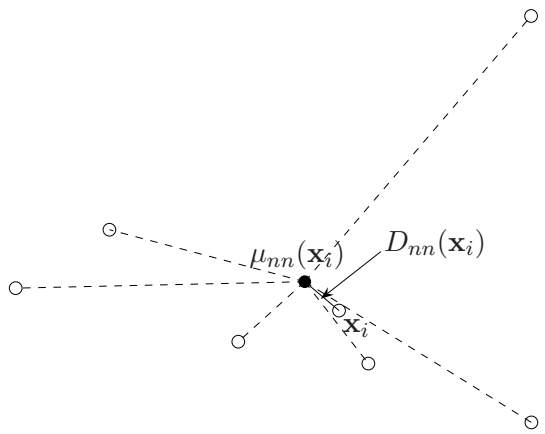


Figure 8.3: The distance $D_{nn}(\mathbf{x}_i)$ between an object $\mathbf{x}_i$ and the mean of its nearest neighbours.

The Euclidean distance matrix D, is not, in general, positive definite ($\mathbf{z}^T D\mathbf{z} \ge 0, \quad \forall \mathbf{z} \in \mathbb{R}^N$). The positive definiteness, or the negative definiteness, of the matrix D is required by Kuhn-Tucker conditions [Rustagi, 1994] for the quadratic optimisation to converge to the global minimum or maximum. However, several techniques can be used to transform a symmetric matrix to the positive (negative) definite. For example one can apply clipping $\mathcal{D} = Q_p\Lambda_p^{1/2}$, where Λ_p are only the positive eigenvalues, or simply taking the square Hadamard power D^{*2} ($D^{*2} = d_{ij}^2$) of the matrix D and adding a small constant to the diagonal $\mathcal{D} = diag(D^{*2}) + c$ [Gower, 1986].

In the optimisation of (8.2), we are looking for k unlabelled objects $\mathbf{x}$ for which the optimised function $\boldsymbol{\alpha}^T\mathcal{D}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\rho$ is maximum. Such a criterion can be used in general with any type of classifier but is especially suited for the Nearest-Neighbour classifier, since it is based on distance relations between objects. To have comparable measurements $\mathcal{D}$, $\mathcal{F}$ and $\mathbf{D}_{nn}$ are scaled to the domain $[0, 1]$ by dividing all values by their maximum value on the unlabelled data.

8.1.2 Density-based diversification

For density-based classifiers the diversification criterion can include instead of distances $\mathbf{D}_{nn}$ densities $P(\mathbf{x})$. We can consider simply densities of unlabelled objects, or similar like in `pdc`, the relative difference between densities of labelled and unlabelled objects $\Delta P(\mathbf{x}_i) = P(\mathbf{x}_i|X_u) - P(\mathbf{x}_i|X_t)$, where $\mathbf{x} \in X_u$; see figure 8.4. The quadratic programming optimisation looks similar to the above optimisation for the Nearest-Neighbour classifier, except now the linear term depends on the difference in density estimates.

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \boldsymbol{\alpha}^T \mathcal{D} \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \boldsymbol{\rho} \\ \text{s.t.} \quad & \boldsymbol{\alpha}^T \mathbf{1} = k, \quad 0 \leq \alpha_i \leq 1, \\ & \boldsymbol{\rho} = \mathcal{F} + \Delta \mathbf{P}. \end{aligned} \tag{8.3}$$

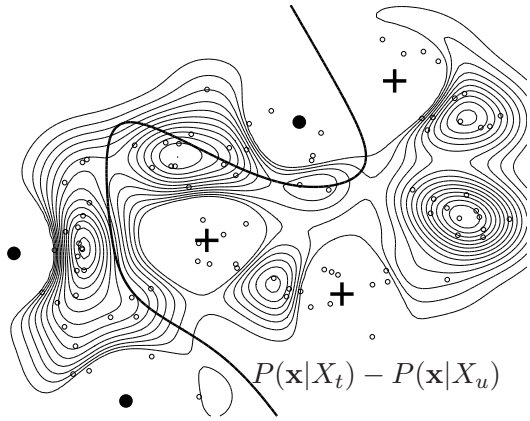


Figure 8.4: The positive difference ΔP in density estimates for labelled $X_t : \{+, \bullet\}$ and unlabelled $X_u : \{\circ\}$ objects plotted as isolines, the current classifier, `Parzen` is drown as a solid, thick line.

Such a method selects a batch of unlabelled objects with large distances $\mathcal{D}$ between selected objects and with the large density value in places where we have no samples yet. Finally the value of the active learning function $\mathcal{F}$ should be also significantly large. In figure 8.4 we can easily point to five unlabelled objects with high value of ΔP indicated by the centres of the concentric isolines. Such objects are remote from each other and are centres of clusters. These make them a potentially informative batch to ask an expert for labels.

Since this diversification method is based on a density estimation it is particularly suitable for density based classifiers e.g. the `Parzen`, `QDA`, `LDA`.

8.1.3 Boundary-based diversification

The last type of a classifier we are considering is the Support Vector Machine (`SVM`). For `SVM` it is convenient to express the mutual label relations of possible labels of unlabelled objects in terms of inner products or similarly the angle between vectors. The angle between two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ can be expressed as follows:

$$\angle(\mathbf{x}_i, \mathbf{x}_j) = \arccos \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|} = \arccos \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) K(\mathbf{x}_j, \mathbf{x}_j)}} \tag{8.4}$$

where $\mathbf{x}_i^T \mathbf{x}_j$ denotes the inner product and K is the between Gram matrix. Similarly to the optimisation (8.2) for `1-NN` we would like to select objects for which the sum of their angles

is maximum and additionally they are centres of clusters in the Hilbert space $\mathcal{H}$. For the Gaussian kernel the denominator in equation (8.4) becomes 1. Since the Gramm matrix K is already positive definite it is easier to minimise the sum of inner products between objects in the batch. Instead of maximising the sum of square angles between selected objects:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \boldsymbol{\alpha}^T K \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \boldsymbol{\rho} \\ \text{s.t.} \quad & \boldsymbol{\alpha}^T \mathbf{1} = k, \quad 0 \leq \alpha_i \leq 1, \\ & \boldsymbol{\rho} = \mathcal{F} + \mathbf{K}_{nn}. \end{aligned} \tag{8.5}$$

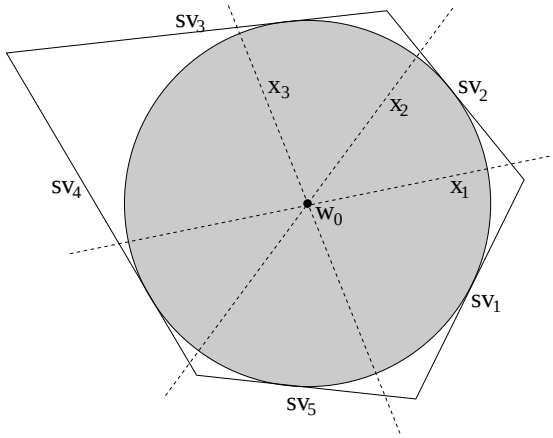


Figure 8.5: Equal division of the approximated version space by three unlabelled objects $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$. **sv** indicates five support vectors and gray circle margin of `SVM`.

where $K_{nn}(\mathbf{x}_i) = \|\mu_{nn}(K(\mathbf{x}_i, :)) - K(\mathbf{x}_i, :)\|$ is the difference between vector $K(\mathbf{x}_i, :)$ and the mean of its neighbours in $\mathcal{H}$.

Figure 8.5 presents a general idea of such a sampling. Let us assume that the problem is linearly separable in the feature space. This means that a version space [Mitchell, 1997] of a particular problem is non-empty. In the case of `SVM`, we can approximate the version space by the support objects and select objects that for two possible labels divide equally such an approximated version space [Tong and Koller, 2000]. Regardless of the true class labels we always reject that half of the classifiers that is inconsistent with the labels of the training data. The tacit assumption is that classifiers are uniformly distributed, i.e. each classifier from the version space is equally probable.

When we consider the selection of a batch of unlabelled objects an informative batch should contain objects that divide the version space equally. The selection of such objects implies that for all their possible labels the size of the version space will be maximally minimised; see figure 8.5. Unlabelled objects $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ divide equally the version space restricted by support vectors $\{\mathbf{sv}_1, \ldots, \mathbf{sv}_5\}$.

8.2 Related work

In this section we shortly explain the difference between our query diversification algorithms and existing methods. In particular, we relate our work to [Brinker, 2003, Park, 2004, Lindenbaum et al., 2004]. These papers present query diversification methods based on various criteria, e.g. similar to the proposed methods, distance between queries [Lindenbaum et al., 2004] or angles between queries in a batch [Brinker, 2003, Park, 2004]. In particular [Lindenbaum et al., 2004] proposed for the `k-NN` rule to construct a batch of unlabelled data using an iterative procedure. At each step a single object is added to a

batch that has a large value of an active learning function and a large distance to already selected objects in a batch. Next, such a constructed batch is presented as a query to an expert. [Brinker, 2003, Park, 2004] discusses similar iterative algorithms for the SVM. However, instead of selecting objects with the maximum sum of distances, they proposed to select objects based on the inner product relations. The algorithms select unlabelled objects that after including them to the training set yield the most orthogonal hyperplanes. A simplified scheme of the existing algorithms is shown below.

$\mathcal{B} = [\,]$
$\mathbf{x}^* = \arg\max_{\mathbf{x} \in X_u} \mathcal{F}(\mathbf{x})$
repeat

1. $\mathcal{B} = \mathcal{B} \cup \{\mathbf{x}^*\}$; $X_u = X_u \backslash \{\mathbf{x}^*\}$
2. $\mathbf{x}^* = \arg\max_{\mathbf{x} \in X_u} [\mathcal{F}(\mathbf{x}) + D(\mathbf{x}, \mathcal{B})]$

until $|\mathcal{B}| = k$

Algorithm 8.1: Standard diversification algorithm.

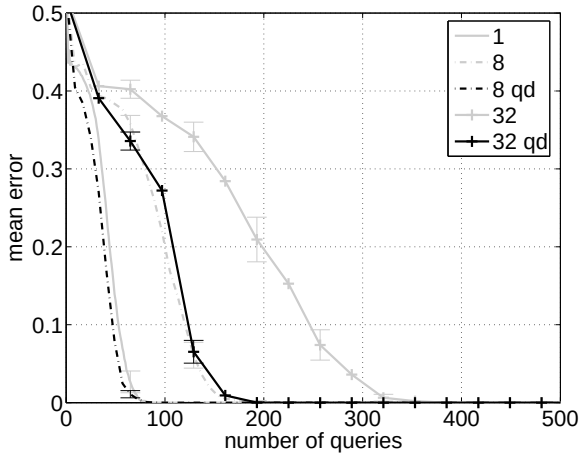
First, an algorithm selects a single unlabelled object with the maximum value of a particular active learning function $\mathcal{F}$. Then the next objects are added to a batch $\mathcal{B}$ for which either the sum of $\mathcal{F}$ and distances to the objects already present in the batch $D(\mathbf{x}, \mathcal{B})$ is large [Lindenbaum et al., 2004] or the inner products are small [Brinker, 2003, Park, 2004]. The process is repeated until the required cardinality of $\mathcal{B}$ is reached.

Because the existing algorithms consider a single candidate to be added to a batch and not an entire batch, they do not necessary select the most informative set of unlabelled objects. The sum of distances and an active learning function do not necessarily reach their maxima for the selected batch. Moreover, the methods presented in these papers maximise distances, or minimise inner products, only between the selected objects; they do not take into account the distribution of unlabelled data. Such methods are sensitive to the presence of outliers, by selecting objects that are far from each other, and not, like in the proposed method, centres of local neighbourhood.

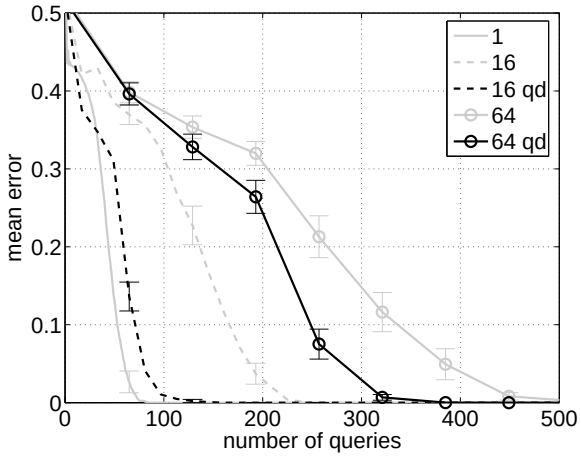
8.3 Experiments

As an illustration, we first test the proposed query diversification algorithms on the artificial *checker board* data set; see figure 8.7. It is a $3D$, two-class data set with an equal number of objects per class. The first class has 13 and the second 14 modes. For clarity, only modes from the visible faces are shown in figure 8.7. This data set, although not realistic to occur in practice, shows clearly the point of the query diversification for active learning methods when multiple queries are to be selected. Moreover, the $2D$ version of this set is widely used as an example in many papers on active learning.

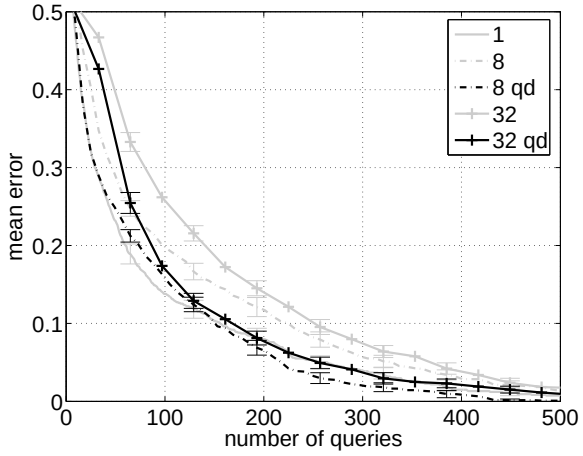
In our experiments, data sets are split according to the information provided in Table 8.1. The initial labelled training set X_t contains two randomly drawn objects per class. The learning proceeds with the queries of $k = \{1, 8, 16, 32, 64\}$ elements. First, in each iteration, a single object is added to the current training set, a query of size $k = 1$, then the learning



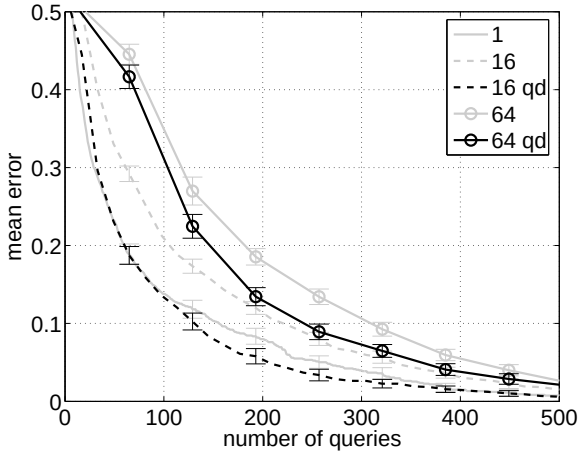
(a) 1-NN $k = \{1, 8, 32\}$



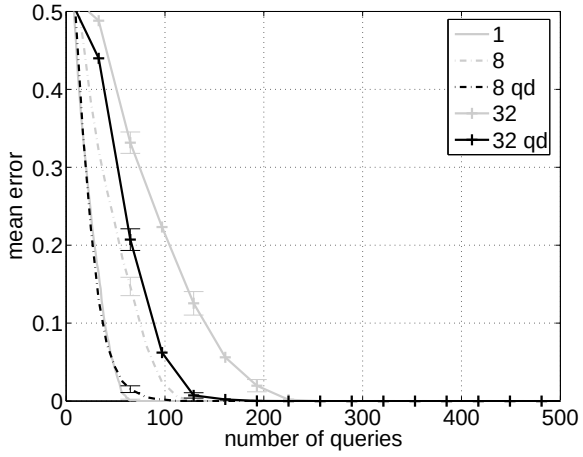
(b) 1-NN $k = \{1, 16, 64\}$



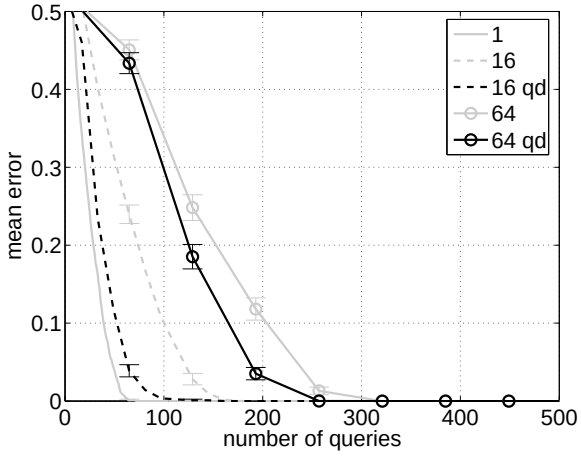
(c) Parzen $k = \{1, 8, 32\}$



(d) Parzen $k = \{1, 16, 64\}$



(e) SVM $k = \{1, 8, 32\}$



(f) SVM $k = \{1, 16, 64\}$

Figure 8.6: Learning curves for the 1-NN, Parzen and ν-SVM for the *checker board* data set. Batches of the sizes $k = \{1, 8, 32\}$ (left) and $k = \{1, 16, 64\}$ (right) are selected according to the uncertainty criterion. The black and gray curves present the error on an independent test set as functions of the training size with and without the query diversification, respectively. The results were averaged over 50 trials.

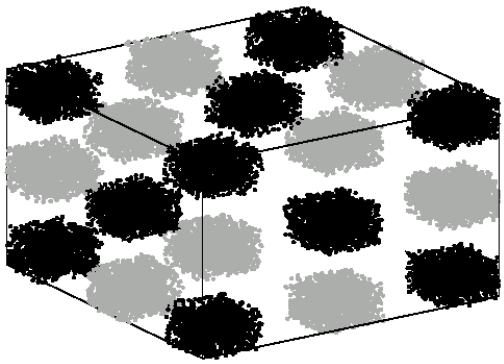


Figure 8.7: 3D *checker board* data set. Only modes from the visible faces are shown.

process is repeated for the query of size $k = 8, 16$ and so on. The learning curve determined for the query based on a single object, $k = 1$, is used as the baseline. The goal is to achieve the performance which is at least as good as obtained for the single object selection algorithm.

Objects are selected according to the uncertainty sampling [Lewis and Gale, 1994] [1]. The learning curves for the `1-NN` rule, the `Parzen` and the ν-`SVM` with a radial-basis kernel are shown in Figure 8.6. The error is measured on an independent test set. The results are averaged over 50 random splits of all data into an initial training set, unlabelled data and a test set. The smoothing parameter of the `Parzen` classifier is optimised according to the maximum likelihood criterion [Duin, 1976] and the ν for ν-`SVM` is set to the `1-NN` leave-one-out error on the training set. In the cases when this error is zero, the ν is set to $\nu = 0.01$. σ in the radial-basis kernel is chosen as the averaged distance to the $\lfloor\sqrt{|X_t|}\rfloor$-nearest neighbour in X_t.

Gray learning curves represent sampling without query diversification and black learning curves present sampling with query diversifications. By observing gray curves it can be seen that by increasing the query size, the number of objects necessary to reach the minimum error classifier increases. This phenomenon is understandable since data are highly clustered and selecting queries based on the active sampling criterion, e.g. the uncertainty sampling, leads to the selection of similar objects from a single mode [2].

The black learning curves in figure 8.6 show the results of the same experiments with the proposed query diversification algorithms, for three types of classifiers for the same batch size. It can be seen that by diversifying queries using the proposed algorithms, the error drops, in this particular learning problem on average about 5% and the difference in the number of queries that is necessary to reach the certain classification error is in average $50 - 100$ in all figures.

Next, we tested the proposed query diversification algorithm on datasets from the UCI Repository [Hettich et al., 1998]. Some information about datasets, such as the size of unlabelled sets and test sets, is presented in table 8.1. Initial training sets consist of two objects

[1] The uncertainty sampling was chosen as an example, however the experimental results are similar for other selective sampling methods, such as `pdc`, `vila`, `qbb` and `MinMax`.

[2] The second observation is that for a two-class problem with an equal number of objects per class, the average error in the beginning of the learning process is larger than 0.5. This is caused by the symmetric mode structure of the dataset itself. Since every mode is surrounded by modes belonging to the other class additional labelled set causes, in the beginning, misclassification of objects from adjacent modes. By increasing the number of clusters, this phenomenon lasts longer.

Table 8.1: The description of data sets used in experiments.

data set	no. of classes	no. of features	size of X_u	size of a test set
checker board	2	3	1000	1500
waveform	3	21	1000	3994
ionosphere	2	34	174	173
sonar	2	60	103	101
diabetes	2	8	382	382
liver	2	6	171	170
ecoli	3	7	134	132
chromo	24	8	500	595
malaysia	20	8	134	117

per class. Because the experiments with all three classifiers and all diversification methods give similar outcomes, we present the results with the ν-SVM and query diversification based on the inner products. The settings of ν and σ are the same as in the experiments with the *checker board* data set. The resulting learning curves for the uncertainty sampling with the query sizes of $k = \{1, 8, 16, 32, 64\}$ are presented in figures 8.8, 8.9 and 8.10. The results are averaged over 50 random splits of data into initial training sets, unlabelled sets and test sets.

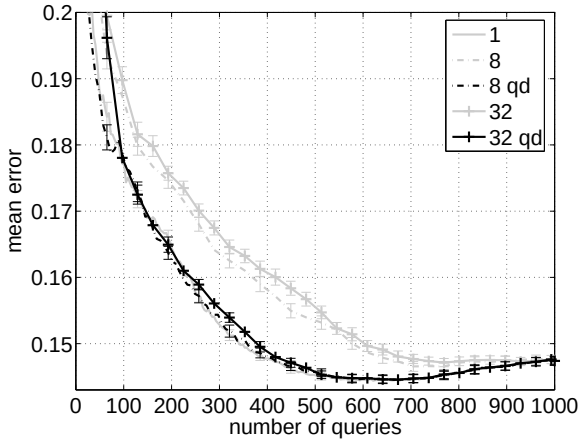
From our experiments, it can be seen that the proposed query diversification algorithm decreases the classification error for the *waveform*, *sonar*, *ecoli*, *liver*, *ionosphere* and *diabetes* data sets. The improvement depend on the batch size. When the size of the batch increases, e.g. when $k = \{16, 32, 64\}$, the performance of the classifier decreases for all data sets. However, when query diversification is applied the performance increases significantly sometimes even outperforming the single query selection algorithm (*sonar* and *diabetes*). When we decrease the batch size to $k = 8$, the classification error is almost comparable with single query selection algorithm.

Since the average classification error is chosen as the performance measure, the observed improvement is related to the number of classes. For data sets with large number of classes such as the *chromo* (24) or *malaysia* (20) sets, it is much more difficult to achieve a better performance, as the error is averaged over more classes. However, even in such challenging cases we can observe an improvement for the *chromo* data set. In case the current classifier is nearly optimal, the presented query diversification algorithm forces a change. This gives a worse or the same classifier. This behaviour can be observed on the *malaysia* dataset.

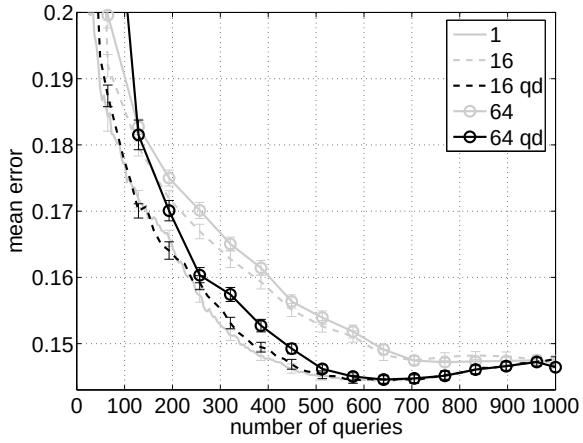
For the *ecoli* and *liver* data sets it is possible to select a smaller subset of the training set on which a classifier performs better as compared to a classifier trained on the entire training set. Such a behaviour of the classification error of the SVM is characteristic for an active selection of training objects for small data sets with a high class overlap; see [Juszczak et al., 2005].

8.4 Conclusions

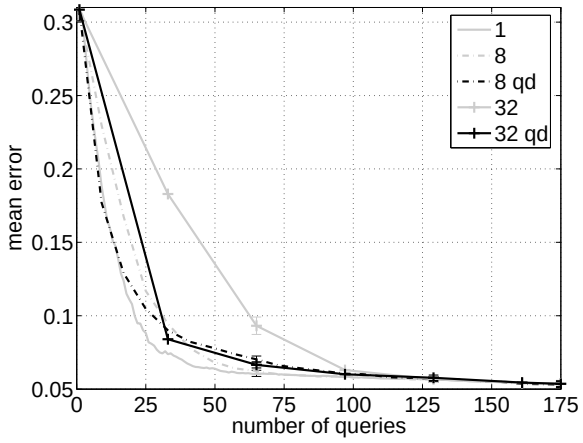
In this section, we have studied the problem of selecting multiple queries in a single draw based on a specified active learning function. In such a selection, a classifier might yield a systematic error by selecting neighbouring objects that contain similar class information. Because of that, the learner should consider not only a particular active learning function but also investigate the influence of retrieving a label of an unlabelled object on other classification labels of



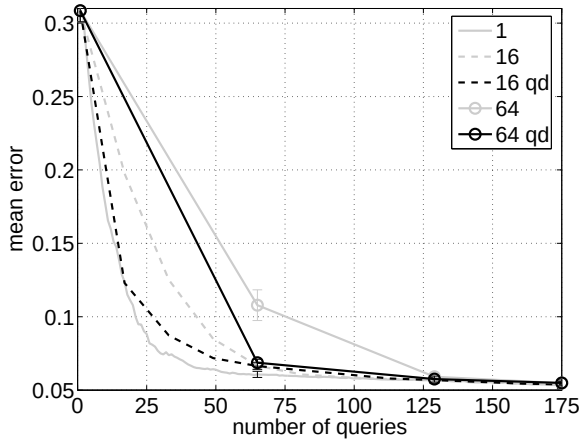
(a) *waveform* $k = \{1, 8, 32\}$



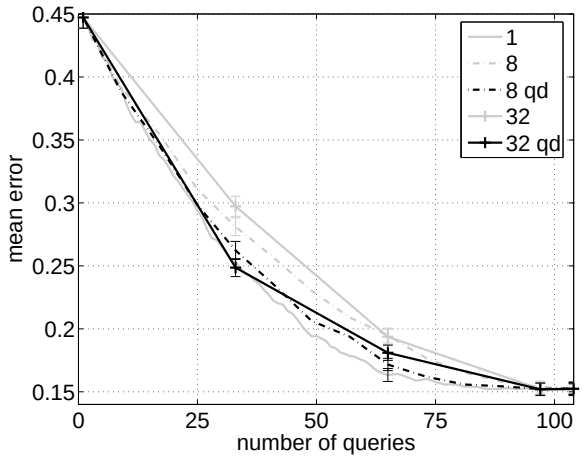
(b) *waveform* $k = \{1, 16, 64\}$



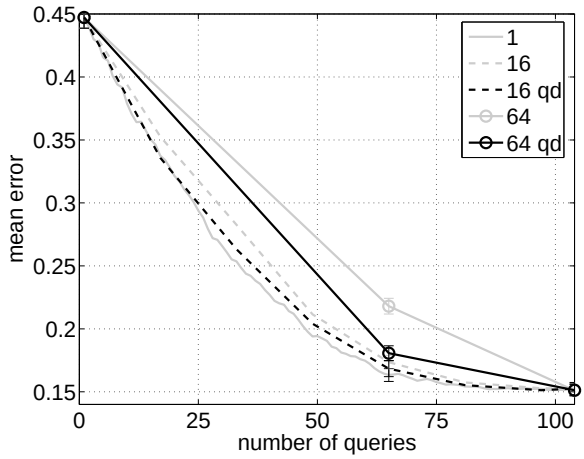
(c) *ionosphere* $k = \{1, 8, 32\}$



(d) *ionosphere* $k = \{1, 16, 64\}$

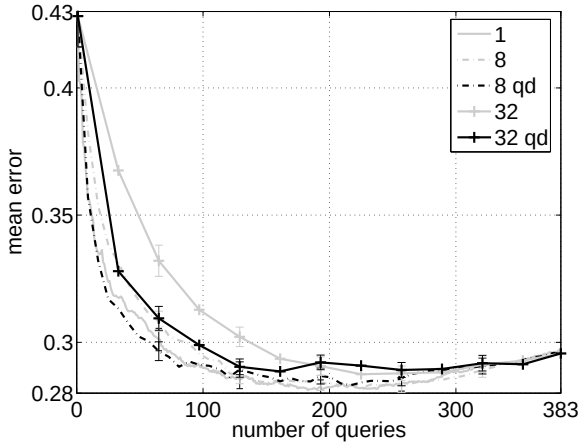


(e) *sonar* $k = \{1, 8, 32\}$

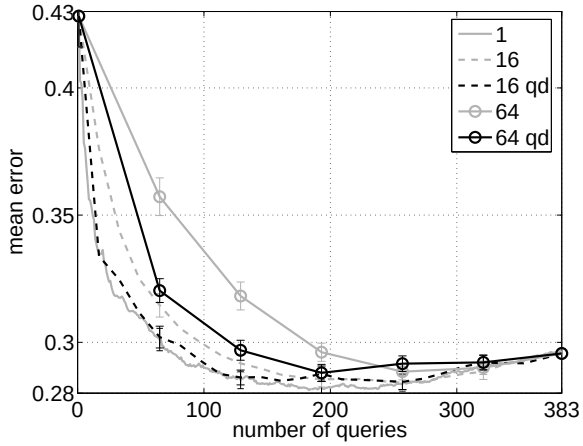


(f) *sonar* $k = \{1, 16, 64\}$

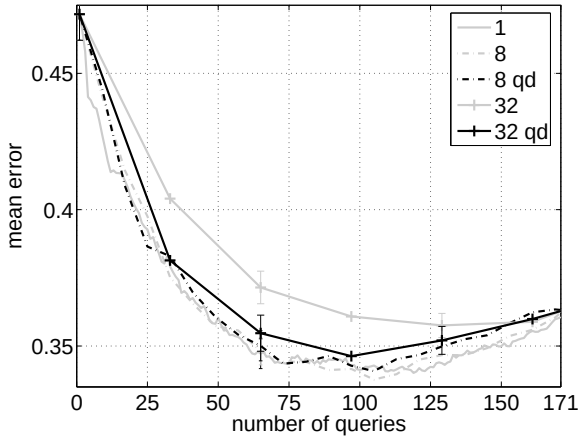
Figure 8.8: Learning curves for the UCI Repository datasets with the query sizes of $k = \{1, 8, 16, 32, 64\}$ for the uncertainty sampling approach with (black) and without (gray) query diversification algorithm. The results are averaged over 50 trails.



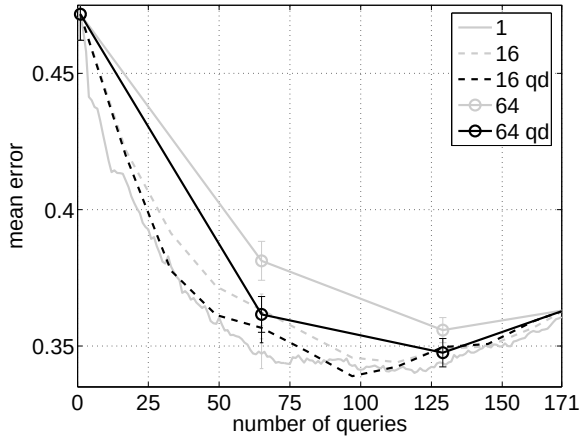
(a) *diabetes* $k = \{1, 8, 32\}$



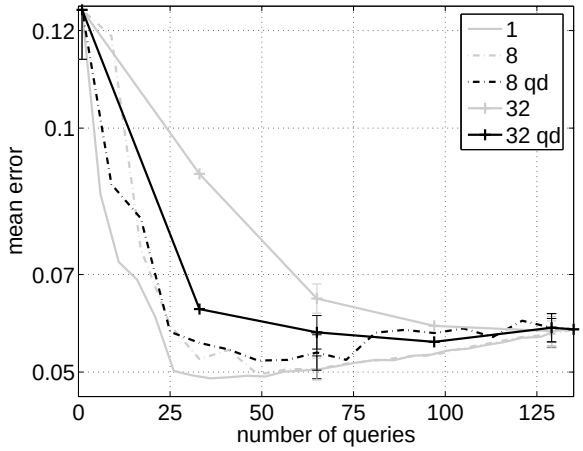
(b) *diabetes* $k = \{1, 16, 64\}$



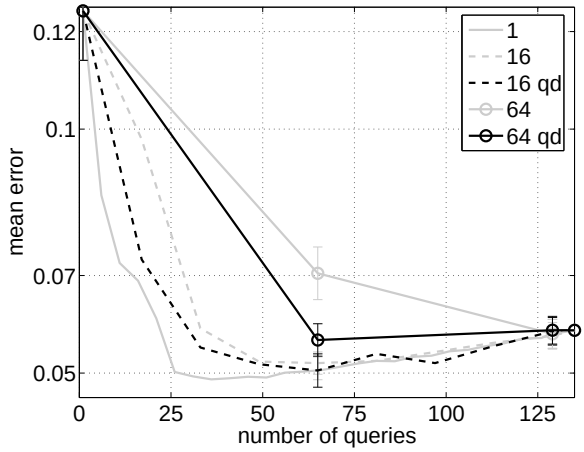
(c) *liver* $k = \{1, 8, 32\}$



(d) *liver* $k = \{1, 16, 64\}$

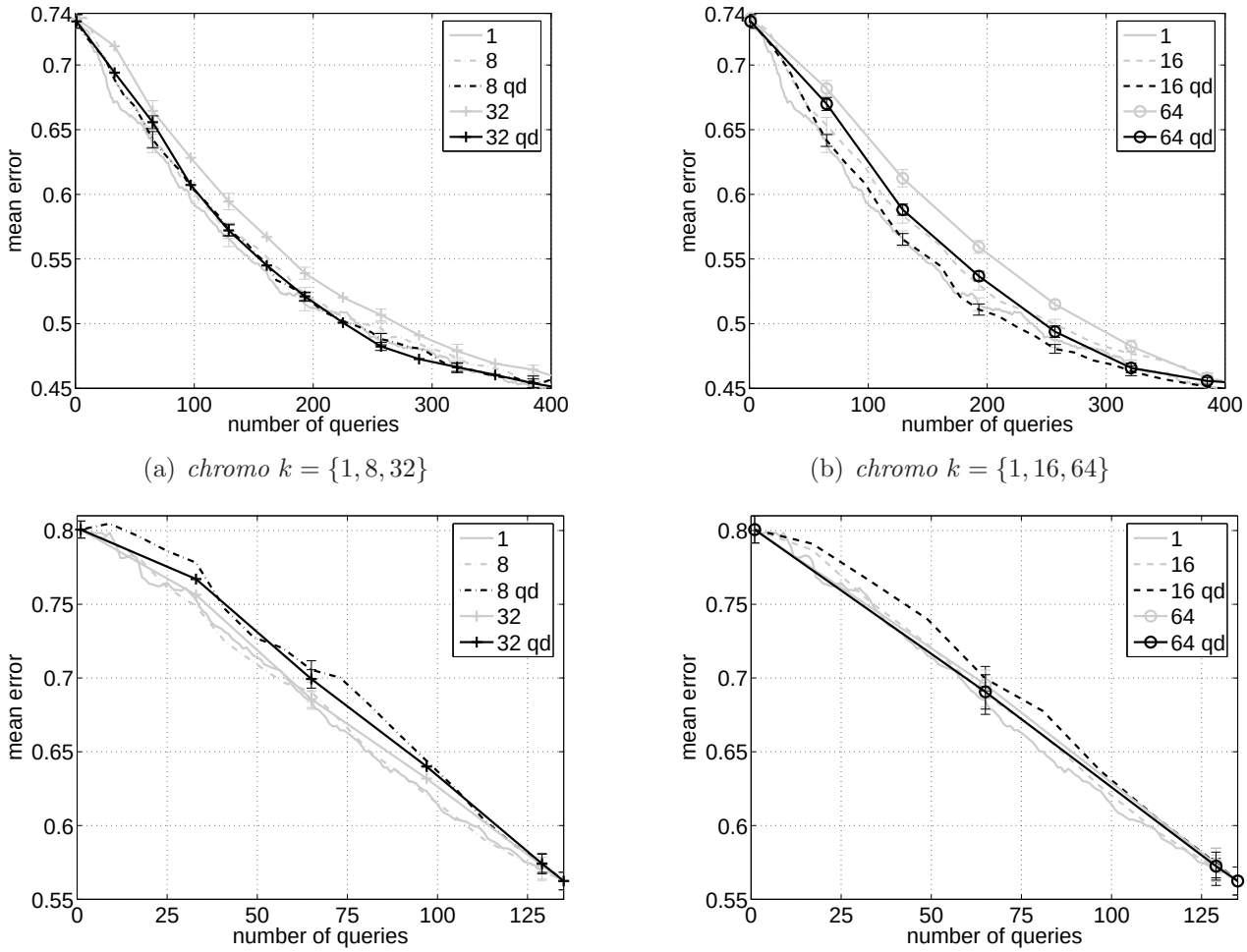


(e) *ecoli* $k = \{1, 8, 32\}$



(f) *ecoli* $k = \{1, 16, 64\}$

Figure 8.9: Learning curves for the UCI Repository data sets with the query sizes of $k = \{1, 8, 16, 32, 64\}$ for the uncertainty sampling approach with (black) and without (gray) query diversification algorithm. The results are averaged over 50 trails.



(a) *chromo* $k = \{1, 8, 32\}$

(b) *chromo* $k = \{1, 16, 64\}$

(c) *malaysia* $k = \{1, 8, 32\}$

(d) *malaysia* $k = \{1, 16, 64\}$

Figure 8.10: Learning curves for the UCI Repository data sets with the query sizes of $k = \{1, 8, 16, 32, 64\}$ for the uncertainty sampling approach with (black) and without (gray) query diversification algorithm. The results are averaged over 50 trails.

potential candidates to a batch. We have formulated the problem of query diversification by using a convex quadratic programming optimisation technique. Different types of classifiers need different queries to reach the same classification error for a given size of a training set. Because of that, the presented algorithm uses properties of the individual classifier type to derive the objective criterion to select batches of queries. Moreover, comparing to the existing iterative procedures we take into account the distribution of labelled and unlabelled data which prevent from selecting outliers.

Chapter 9

Semi-supervised learning

Imagine a classification problem where it is known that a distribution of classes changes. However, there is no possibility to ask an expert for additional labels to update a classifier, as it is the case in active learning. For example, in geological problems a classifier is trained on examples of minerals collected from various places. Next, the classifier is applied in a new location to classify some new geological samples. It is expected that the distribution of classes changes. A simple example of such shifted distributions is shown in figure 9.1.

From the figures it can be observed that the distribution and domains of classes estimated on a training set X_t, figure 9.1(b), are different from the ones estimated for a test set X_u, figure 9.1(c). One of the fundamental assumptions of machine learning is not fulfilled, namely that a training set comes from the same distribution as a test set. Moreover, even an infinite training set, sampled i.i.d., does not necessary give a smaller error on a test set, in this situation. However, by considering both: class information from a training set and a distribution of unlabelled objects could significant improve the classification performance. The problem that we are facing is called classification with partially labelled data or semi-supervised learning [Juszczak and Duin, 2005].

The problem of classification with partially labelled data requires linking the unlabelled input distribution $P(\mathbf{x})$ with the conditional distribution $P(\omega|\mathbf{x})$ obtained from the labelled data. The latter should, for example, vary little in high density regions. The key problem is to articulate a general principle behind this and other such reasonable assumptions. In this section, we provide a new approach to semi-supervised learning, based on the `EM`-algorithm, to estimate labels for the unlabelled dataset. The presented method does not require clustering assumptions and the approach remains tractable even for continuous marginal class densities.

We start with similar assumptions to the ones in active learning i.e. the access to unlabelled objects is easy and there is a significant cost, in time or money, to label additional objects. Therefore, usually we label a small number of objects and hope, that they are sufficiently representative for the classification problem. However, to benefit from the remaining unlabelled objects, one must exploit implicitly or explicitly the link between density $P(\mathbf{x})$ over objects $\mathbf{x}$ and the conditional $P(\omega|\mathbf{x})$ representing the posterior probability of the labels $\boldsymbol{\omega}$. There is no access to an expert who can provide additional labels.

In statistical pattern recognition [Bishop, 1995, Duda et al., 2001] classification methods mostly do not attempt to explicitly model or incorporate information from the density $P(\mathbf{x})$. However, some classification algorithms such as density based algorithms as the `Parzen` classifier [Parzen, 1962, Duda et al., 2001] or transductive `SVM` [Vapnik, 1998] have a possibility to relate $P(\mathbf{x})$ to $P(\omega|\mathbf{x})$; the decision boundary is biased to fall preferentially in low density regions of $P(\mathbf{x})$.

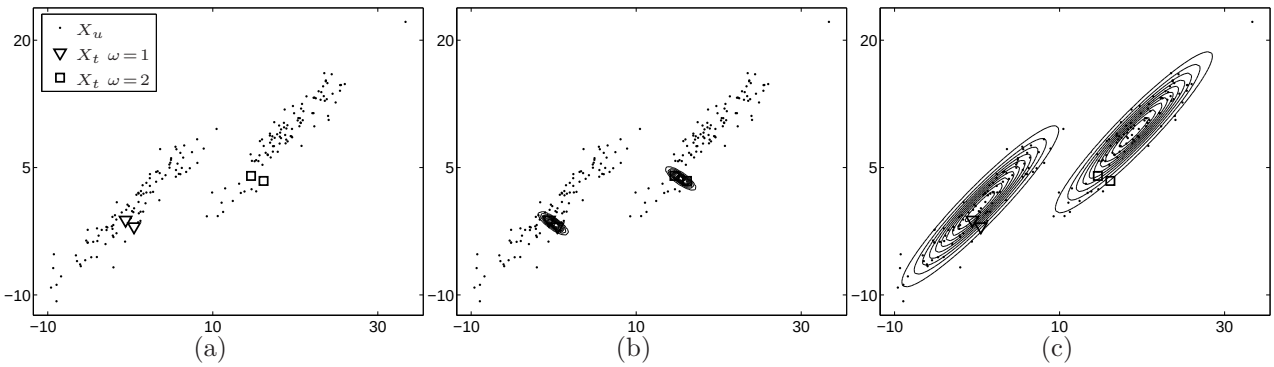


Figure 9.1: (a) Labelled $X_t : \{\nabla, \square\}$ and unlabelled data $X_u : \{\cdot\}$. (b) The density estimate of X_t. (c) The density estimate of X_u.

In such algorithms, the unlabelled objects, e.g a large test set to be classified, provide additional information about the structure of the domain while the few labelled objects identify the classification task expressed in this structure. A tacit assumption in this context is to associate high-density clusters in data with pure classes. When this assumption is appropriate, it is only required to label a single object per cluster to classify the whole dataset.

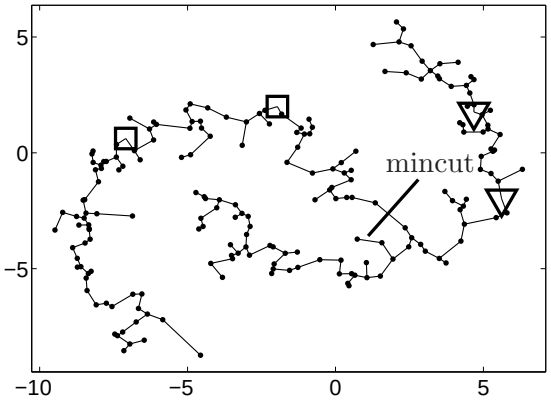


Figure 9.2: The mincut algorithm.

The presented problem is in broad terms related to a number of other problems like maximum entropy discrimination [Jaakkola et al., 1999], data clustering by information bottleneck [Tishby and Slonim, 2000], and minimum-entropy data partitioning [Roberts et al., 2001].
In this section, we investigate label propagation from a small labelled set over a large unlabelled set for density based classifiers in the semi-supervised learning framework, using as an example density-based classifiers e.g. the `Parzen` classifier, the linear discriminant analysis (`LDA`) and quadratic discriminant analysis (`QDA`).

The main difference between the various semi-supervised learning algorithms proposed in literature, such as spectral methods [Chapelle et al., 2002], random walks [Szummer and Jaakkola, 2001], graph mincuts [Blum et al., 2004] and transductive `SVM` [Vapnik, 1998], lies in the way of realising the assumption of the labels consistency. However, the following three assumptions are often made about the representation space where the classification problem is present:

1. nearby objects are likely to have the same label,

2. objects on the same data structure, e.g. a cluster or a manifold, are likely to have the same label,

3. the decision boundary should lie in regions of low density[1]. For example in handwritten digit recognition where one tries to classify e.g. digits 2 and 5, one expects that the

[1]This assumption is related to the previous one.

probability of having a digit which is between class 2 and class 5 is lower than the probability of a distinct 2 and 5.

An example of an existing semi-supervised method is the mincut algorithm [Blum et al., 2004]. The method computes an undirected graph between labelled and unlabelled objects; see figure 9.2. The weights in the graph are related to distances and label relations between objects. Next the graph is cut using a minimum energy criterion.

The semi-supervised learning method proposed in this section is based on the stability of estimated labels for unlabelled objects. In contradiction to the mentioned methods, in particular [Chapelle et al., 2002, Blum et al., 2004, Szummer and Jaakkola, 2001], there is not an implicit clustering step involved in the label propagation process. Therefore, there is no necessity to specify or optimise the number of clusters beforehand.

9.1 Semi-supervised density based algorithms

Given a partially labelled data set $X = \{(\mathbf{x}_1, \omega_1), \ldots, (\mathbf{x}_n, \omega_n), \mathbf{x}_{n+1}, \ldots, \mathbf{x}_{n+m}\} \subset \mathbb{R}^N$, the first n objects are labelled X_t and the remaining m objects $\mathbf{x}_i \in X_u$ $(n+1 \leq i \leq n+m)$ are unlabelled. The goal is to predict labels of the unlabelled objects. An example of such a problem has been presented in figure 9.1. Our classification model assumes that each data example has a crisp label for $\mathbf{x}_i \in X_t$, or a distribution $P(\omega^{(j)}|\mathbf{x}_i)$ for $\mathbf{x}_i \in X_u$, over the class labels[2]. These distributions are unknown and represent the parameters to be estimated. Given an object $\mathbf{x}_i$, which may be labelled or unlabelled, we interpret the probability that $\mathbf{x}_i$ has a label $\omega^{(j)}$ as a weighted combination of crisp and soft labels of all objects in $X \in [X_t, X_u]$. First, we show how to incorporate soft labels into various density based classifiers. Next we present an algorithm to estimate soft labels for $\mathbf{x}_i \in X_u$. Finally, we discuss the optimisation of the parameters of the classifier on both labelled and unlabelled data.

Assume that sets of labels $P(\omega^{(j)}|\mathbf{x}_i) = \{0, 1\}$, $\forall_{\omega^{(j)} \in \boldsymbol{\omega}}$, $\forall_{\mathbf{x}_i \in X_t}$, $i = 1, \ldots, n$ and $0 \leq P(\omega^{(j)}|\mathbf{x}_i) \leq 1$, $\forall_{\omega^{(j)} \in \boldsymbol{\omega}}$, $\forall_{\mathbf{x}_i \in X_u}$, $i = 1, \ldots, m$ is given. Our task is to predict label of a new object $\mathbf{x}_k$ using labelled and unlabelled data.

9.1.1 Semi-supervised linear discriminant analysis (*soft*-LDA)

The discriminant function $h_j(\mathbf{x})$ [Duda et al., 2001] is computed by the linear discriminant analysis as follows:

$$h_j(\mathbf{x}_k) = (\Sigma^{-1}\mu_j)^T \mathbf{x}_k^T - \frac{1}{2}\mu_j^T \Sigma^{-1} \mu_j + \ln P(\omega^{(j)}) \tag{9.1}$$

We can replace the mean μ_j and the probability $P(\omega^{(j)})$ of a class by their weighed versions $\tilde{\mu}_j$ and $\tilde{P}(\omega^{(j)})$ obtaining the semi-supervised version of LDA *soft*-LDA.

$$\tilde{\mu}_j = \frac{1}{n+m} \sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i)\mathbf{x}_i, \qquad \tilde{P}(\omega^{(j)}) = \frac{1}{n+m} \sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i), \qquad \mathbf{x}_i \in \{X_t, X_u\} \tag{9.2a}$$

$$\Sigma_{j_{kq}} = \frac{1}{n+m} \sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i)(x_{i_l} - \tilde{\mu}_{j_l})(x_{i_q} - \tilde{\mu}_{j_q}), \quad \Sigma = \frac{1}{C} \sum_{j=1}^{C} \Sigma_j, \qquad \mathbf{x}_i \in \{X_t, X_u\} \tag{9.2b}$$

[2] $P(\omega^{(j)}|\mathbf{x}_i)$ are also called soft labels

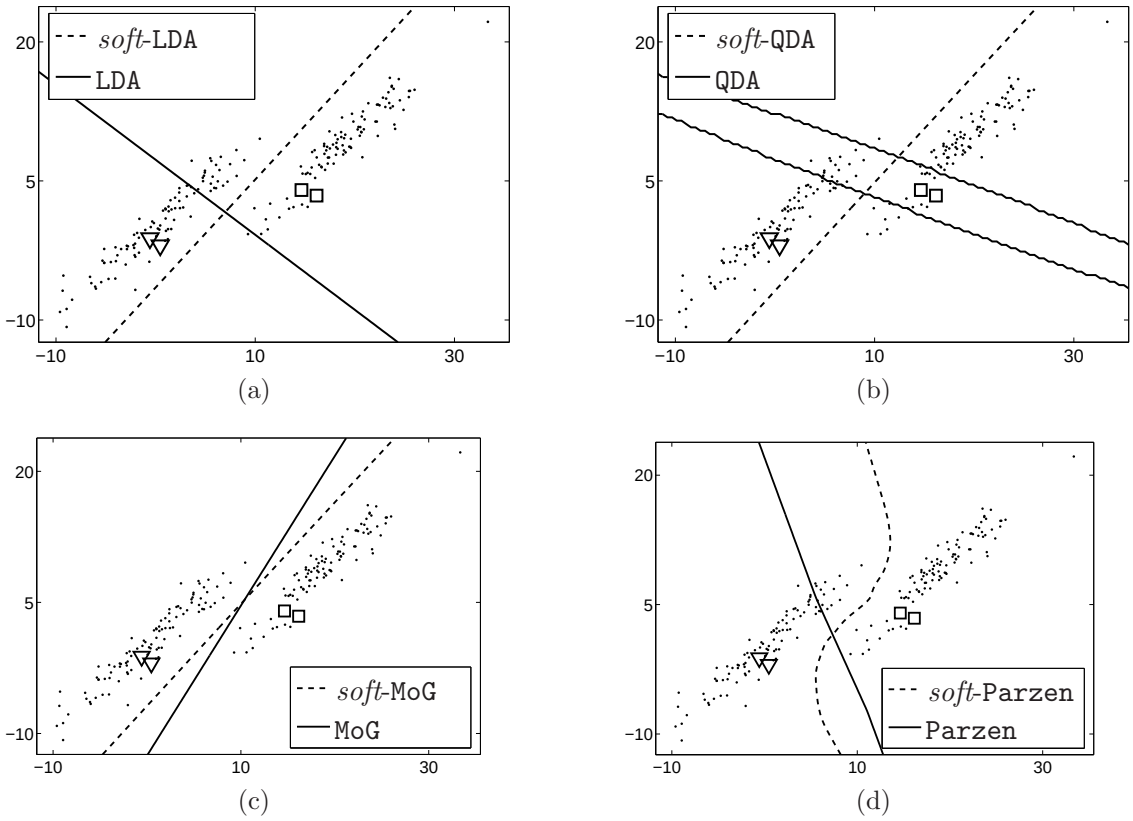


Figure 9.3: Various standard density based classifiers trained on labelled set and their 'soft' versions that considers also a distribution of unlabelled data.

where $1 \leq [l, q] \leq N$ run over feature indices of $\mathbb{R}^N$. We assume n labelled and m unlabelled objects. $p(\mathbf{x}_k|\omega^{(j)})$ is obtained by scaling $h_j(\mathbf{x})$ by a sigmoidal transformation and $p(\omega^{(j)}|\mathbf{x}_k)$ from the Bayes rule. By scaling $p(\omega^{(j)}|\mathbf{x}_k)$, such that $\sum_{j=1}^{C} p(\omega^{(j)}|\mathbf{x}_k) = 1$, we obtain $P(\omega^{(j)}|\mathbf{x}_k)$ which can be included in the above formulae and the *soft*-LDA can be retrained including the test object $\mathbf{x}_k$. Note that the covariance matrix Σ is averaged over classes and thereby depend on class labels. An example of LDA and *soft*-LDA is shown in figure 9.3(a).

9.1.2 Semi-supervised quadratic discriminant analysis (*soft*-QDA)

The discriminant function $h_j(\mathbf{x})$ [Duda et al., 2001] is computed by the quadratic discriminant analysis as follows:

$$h_j(\mathbf{x}_k) = -\frac{1}{2}\mathbf{x}_k^T\Sigma_j^{-1}\mathbf{x}_k + (\Sigma_j^{-1}\mu_j)^T\mathbf{x}_k^T - \frac{1}{2}\mu_j^T\Sigma_j^{-1}\mu_j - \frac{1}{2}\ln\det(\Sigma_j) + \ln P(\omega^{(j)}) \tag{9.3}$$

The mean μ_j and the probability $P(\omega^{(j)})$ are computed similarly to equation (9.2a). However, the covariance matrix is now computed per class. To incorporate soft labels into the computation of the covariance matrix we weight it by the posterior probability:

$$\Sigma_{j_{kq}} = \frac{1}{n+m}\sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i)(x_{i_l} - \tilde{\mu}_{j_l})(x_{i_q} - \tilde{\mu}_{j_q}), \qquad \mathbf{x}_i \in \{X_t, X_u\} \tag{9.4}$$

where $1 \leq [l, q] \leq N$ run over feature indices of $\mathbb{R}^N$. The example of QDA and *soft*-QDA is shown in figure 9.3(b).

9.1.3 Semi-supervised mixture of Gaussians (*soft*-MoG)

In the mixture of Gaussians, with γ mixtures, The discriminant function $h_j(\mathbf{x})$ [Duda et al., 2001] is computed by:

$$h_j(\mathbf{x}_k) = \ln \sum_{c=1}^{\gamma} \alpha_c \frac{1}{\sqrt{\det(\Sigma_{cj})}} \exp\left[-(\mu_{cj} - \mathbf{x}_k)^T \Sigma_{cj} (\mu_{cj} - \mathbf{x}_k)\right] + \ln P(\omega^{(j)}) \tag{9.5}$$

where α_c is the mixing component of the c-th mixture. The mean μ_{cj} and covariance matrix Σ_{cj} are computed similarly to equations (9.2a) and (9.4), however now only objects that belong to the c-th mixture are considered. The example of a decision function computed by MoG and *soft*-MoG is shown in figure 9.3(c).

9.1.4 Semi-supervised Parzen Window classifier (*soft*-Parzen)

The last classifier we discuss here is the Parzen Window classifier. The probability that object $\mathbf{x}_k$ belongs to a class $\omega^{(j)}$ is computed as a weighted sum of all objects in the labelled and unlabelled sets:

$$h_j(\mathbf{x}_k) = \sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i)\ \varphi\left(\frac{\mathbf{x}_i - \mathbf{x}_k}{\sigma}\right) \tag{9.6}$$

where $\varphi(\frac{\mathbf{x}_i - \mathbf{x}_k}{\sigma})$ is the kernel function computed between all objects. In the standard Parzen Window classifier $P(\omega^{(j)}|\mathbf{x}_i) = [0, 1]$ and in the semi-supervised version of it $P(\omega^{(j)}|\mathbf{x}_i) = [0, 1]$ for $\mathbf{x}_i \in X_t$ and $0 \leq P(\omega^{(j)}|\mathbf{x}_i) \leq 1$ for $\mathbf{x}_i \in X_u$. The example of a decision function computed by Parzen and *soft*-Parzen is shown in figure 9.3(d).

In the remaining part of this section we consider the *soft*-Parzen as an example. The proposed method can be applied in a straightforward way to the other considered classifiers.

Estimation of soft labels $P(\omega^{(j)}|\mathbf{x}_i)$

So far we have assumed that soft labels $P(\omega^{(j)}|\mathbf{x}_i)$ are given. However, in general, the $P(\omega^{(j)}|\mathbf{x}_i)$ are only available for labelled objects X_t and have to be estimated for unlabelled objects X_u. We propose to estimate $P(\omega^{(j)}|\mathbf{x}_i)$ using the conditional maximum log-likelihood as the criterion. $P(\omega^{(j)}|\mathbf{x}_i)$ is estimated for unlabelled objects for fixed values of the parameters of the classifier e.g. a number of mixtures in MoG. For Parzen the objective function can be written as:

$$\max_{P(\omega^{(j)}|\mathbf{x}_i)} \quad \sum_{j=1}^{C} \ln \sum_{i=1}^{n+m} P(\omega^{(j)}|\mathbf{x}_i) \varphi\left(\frac{\mathbf{x}_i - \mathbf{x}_k}{\sigma}\right) \tag{9.7}$$

where $P(\omega^{(j)}|\mathbf{x}_i) = [0, 1]$ for labelled objects and $P(\omega^{(j)}|\mathbf{x}_i) = 0$ for unlabelled objects at the beginning of a optimisation. Since $\varphi(\frac{\mathbf{x}_i - \mathbf{x}_k}{\sigma})$ are fixed this objective function is jointly convex in the free parameters and has a unique maximum value. This convexity also guarantees that this optimisation is easily performed via the EM-algorithm. We can write similar functions for *soft*-LDA, *soft*-QDA and *soft*-MoG.

Estimation of the hyperparameters for semi-supervised algorithms

In the previous subsection we have assumed that the parameters of the classifier were known and fixed. In this section, we compute and optimise parameters for the set of labelled and

unlabelled objects in the maximum likelihood sense for the known and fixed sets of soft labels $P(\omega^{(j)}|\mathbf{x}_i)$. The examples of parameters that we optimise are: a regularisation parameter in `LDA` and `QDA`, the number of mixtures in `MoG` and in the `Parzen` classifier it is a type and smoothing parameter of a kernel should be chosen. Several functions can be used as a kernel $\varphi(\frac{\mathbf{x}_i-\mathbf{x}_k}{\sigma})$ e.g.:

$$\begin{aligned}
&\text{uniform} \quad \frac{1}{2}\varphi(\|u\| \leq 1),\\
&\text{triangle} \quad (1-\|u\|)\varphi(\|u\| \leq 1),\\
&\text{Epanechnikov} \quad \frac{3}{4}(1-u^2)\varphi(\|u\| \leq 1),\\
&\text{cosinus} \quad \frac{\pi}{4}\cos(\frac{\pi}{2}u)\varphi(\|u\| \leq 1).
\end{aligned}$$

where $u = \frac{\mathbf{x}_i-\mathbf{x}}{\sigma}$. Although the choice of a kernel function is open, it should be a proper density estimator [Fukunaga, 1972]. We optimised σ based on a leave-one-out maximum likelihood estimation [Duin, 1976].

Proposed algorithm

The proposed algorithm for semi-supervised learning is based on the optimisation of soft labels, according to equation (9.7), for a given σ. The initial estimate σ_t of σ is optimised for just the labelled objects $\mathbf{x}_i \in X_t$. The final σ_{tu} is optimised for both labelled and unlabelled objects $\mathbf{x}_i \in [X_t, X_u]$. In a series of k `EM`-algorithms σ takes the values:

$$\sigma_t > \sigma_2 > \ldots > \sigma_{k-1} > \sigma_{tu}$$

The change in σ from large, σ_t, to small, σ_{tu}, values, during optimisation of soft labels, changes the stress between global label consistency and the local label consistency.

The proposed algorithm of classification with a partially labelled dataset is summarised in Algorithm 9.1. In the initial step of the algorithm the soft labels are computed using only labelled objects, $P(\omega^{(j)}|\mathbf{x}_i) = 0 \; \forall \mathbf{x}_i \in X_u$. In the second step based on the current estimation of σ_t soft labels are optimised $P(\omega^{(j)}|\mathbf{x}_i)$, $\forall \mathbf{x}_i \in X_u$ using the maximum likelihood criterion. Next, equation (9.6) is recomputed using both crisp and soft labels. Step 3 is repeated until the difference between the current estimated labels $P_t(\omega^{(j)}|\mathbf{x}_i)$ and the previous estimated labels $P_{t-1}(\omega^{(j)}|\mathbf{x}_i)$ is smaller than ϵ. The procedure is repeated for k different σ-s.

1. Set the number of `EM`-algorithms to k; compute $\sigma_1 = \sigma_l$ and $\sigma_k = \sigma_{lu}$; set $t = 1$ and a stopping criterion ϵ;

2. Compute: $\sigma_1 > \sigma_2 > \ldots > \sigma_{k-1} > \sigma_k$ for each `EM`-algorithm;
   set $P_0(\omega^{(j)}|\mathbf{x}_i) = 0, \quad \forall \mathbf{x}_i \in X_u, \forall \omega^{(j)} \in \boldsymbol{\omega}$

while $t \leqslant t_{max}$

3. Optimise soft labels $P_t(\omega^{(j)}|\mathbf{x}_i)$ based on equation (9.7) with a fixed σ_t; using the soft labels $P_{t-1}(\omega^{(j)}|\mathbf{x}_i)$ from step $t-1$ as the initialisation of the labels;

4. Repeat 3 until the stopping criterion is reached
   e.g.$\sum_j^C \sum_i^m |P_t(\omega^{(j)}|\mathbf{x}_i) - P_{t-1}(\omega^{(j)}|\mathbf{x}_i)| < \epsilon$; $t = t+1$;

end

Algorithm 9.1: *soft*-`Parzen`.

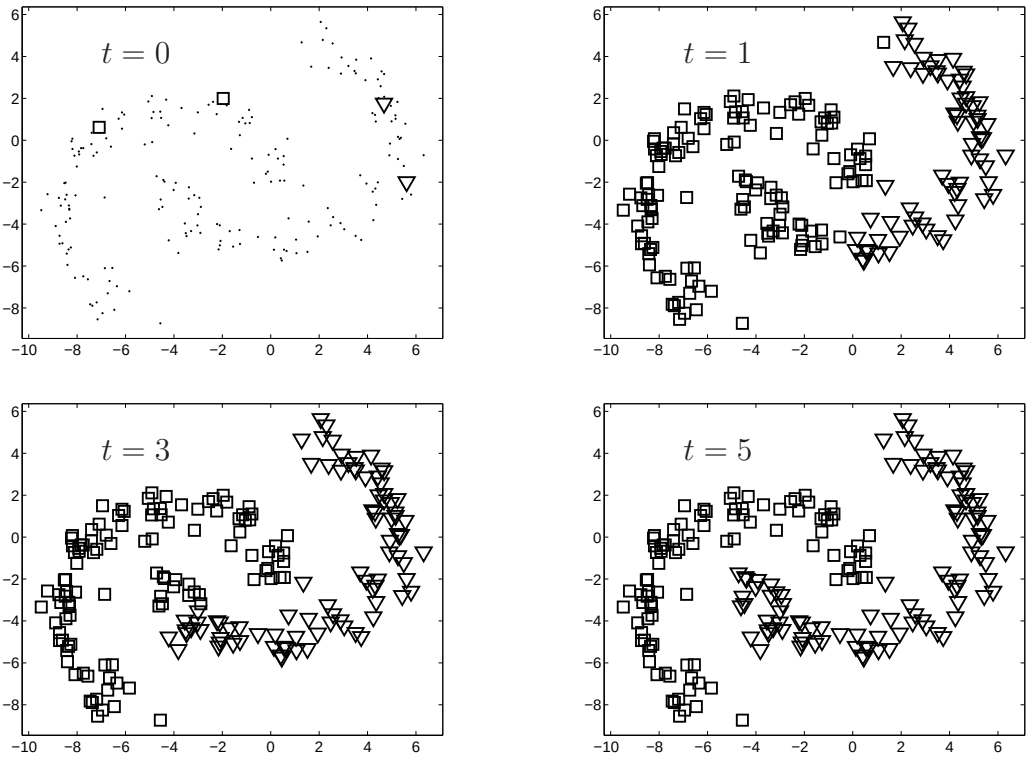
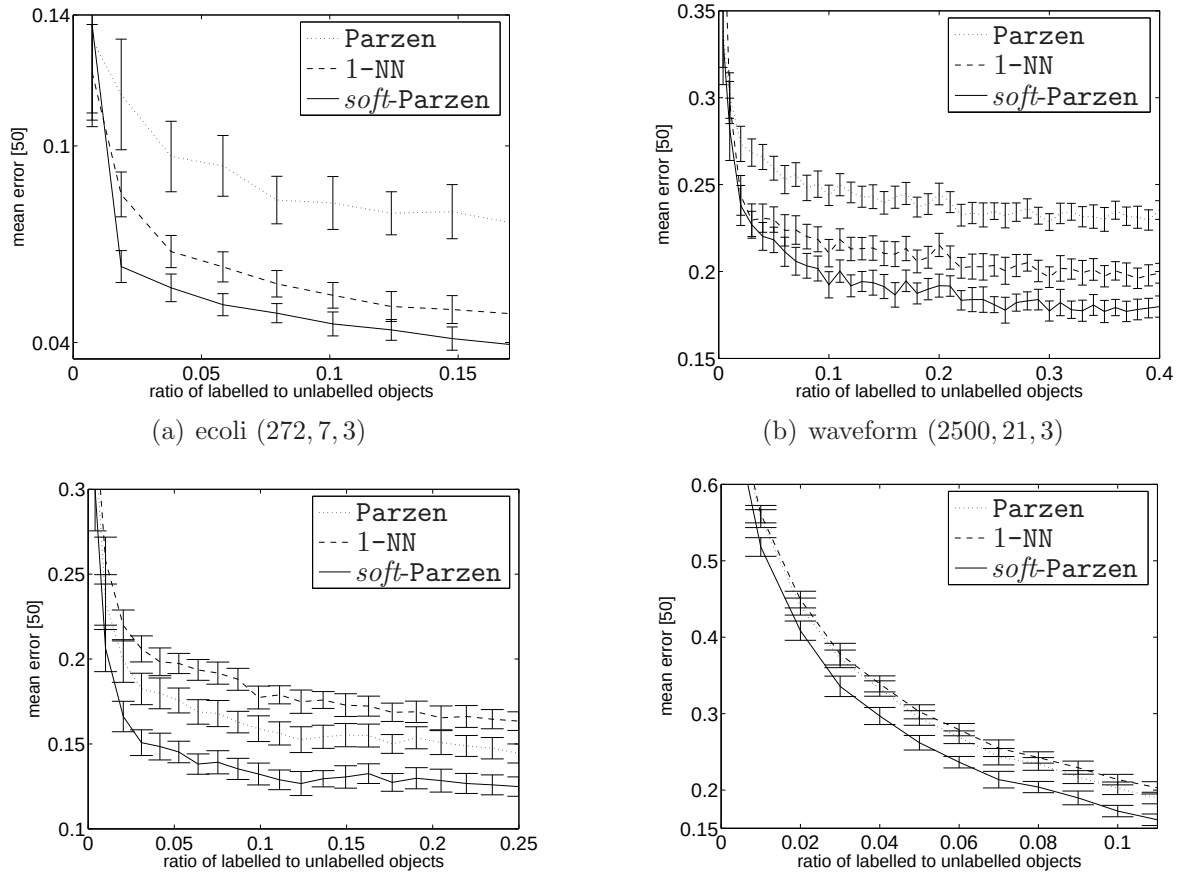


Figure 9.4: Label estimates for the *soft*-`Parzen` algorithm for the banana shape dataset. Labelled (soft and crisp labels) objects are denoted by $\{\nabla, \square\}$ and unlabelled objects are denoted by $\{\cdot\}$.

9.2 Experiments

Consider an example, figure 9.4, of classification with the proposed algorithm. We are given 2 labelled objects per class and 196 unlabelled objects in an intertwining two banana shape patterns. This pattern has a manifold structure where distances are locally but not globally Euclidean, due to the curved arms. Therefore, the pattern is difficult to classify for traditional algorithms using locally defined relations, such as 1-nearest neighbour. We used the proposed algorithm, described in Algorithm 9.1, to incorporate unlabelled data into the Parzen density estimator and scale the Euclidean distance between objects using their soft labels. Figure 9.4 shows three different timescales. At $t = 1$, σ is overestimated, therefore there are large Gaussian clusters, and the $P(\omega^{(j)}|\mathbf{x}_i)$'s are only estimated roughly. At $t = 3$, σ becomes smaller and local label relations in marginal regions start to change the soft labels. At $t = 5$ almost all objects, apart of one, have correct labels.

Next, we evaluate the performance of the presented algorithm on some of the UCI repository datasets [Hettich et al., 1998]: *waveform*, *satellite*, *letter*, *ecoli*. Datasets were divided into two parts: labelled set X_t and the unlabelled set X_u constituted from remaining objects, the ratio $\frac{X_t}{X_u}$ is indicated by numbers on the abscissa. The label propagation was performed on X_u and the obtained classifier was tested on the same set of unlabelled data X_u. The random division was repeated 50 times for each ratio $\frac{X_t}{X_u}$. The performance of the proposed algorithm (*soft*-`Parzen`) is compared with the 1-nearest neighbour classifier (`1-NN`) and the `Parzen` classifier trained just on the labelled objects (`Parzen`). The mean error and the standard deviation are shown in figure 9.5. It can be seen, that the proposed *soft*-`Parzen` algorithm outperforms



(a) ecoli $(272, 7, 3)$ (b) waveform $(2500, 21, 3)$

(c) satellite $(1500, 36, 6)$ (d) letter $(2000, 16, 26)$

Figure 9.5: Mean square error and standard deviation for *soft*-`Parzen` compared with a classifier trained just on a labelled dataset `Parzen` and `1-NN` for UCI Repository datasets: *ecoli*, *waveform*, *satellite*, *letter*. Numbers in brackets indicate the size of datasets, the number of features and classes in a dataset.

both: `1-NN` and the `Parzen` classifier trained on just labelled objects, on the considered classification problems. In case of *waveform* and *ecoli* the performance of *soft*-`Parzen` is close to `1-NN` and for *satellite* and *letter* there is significant improvement.

The *soft*-`Parzen` and `1-NN` perform similar if distances between objects in pure clusters and between clusters differ significantly. However, if there is no clear cluster structure in the data the *soft*-`Parzen` might outperform the `1-NN` significantly.

The performance of the proposed method depends on the quality of the labelled data and their relation to the structure of the unlabelled dataset. If the clusters of unlabelled data are not related to the class information, it is hard to expect that the proposed method performs well. For a broader discussion about merits and disadvantages of the semi-supervised learning we point the reader to the paper [Cohen et al., 2004].

9.3 Conclusions

The proposed algorithm based on expectation maximisation of soft labels *soft*-`Parzen` provides a robust variable resolution approach to classifying data sets with significant cluster structure

and very few labels. When the cluster structure is absent or unrelated to the classification task, the proposed method can be expected to give small or non improvement over a classifier trained just on the labelled dataset. In such cases the performance is strongly related to the quality of the already labelled set.

Chapter 10

Conclusions

In this thesis, we have investigated the problem of one-class classification and the problem of enhancing the classification performance by the use of unlabelled data. We have tried to answer the following questions:

1. Can a classifier be constructed, trained and evaluated based on samples from just a single class?

2. What should be the measure to minimise the chance of accepting non-target or outlier objects in one-class classification problems?

3. How can we efficiently ask experts for new labelled examples?

4. How can we combine the knowledge about $p(\mathbf{x}|\omega)$ from a small labelled set with $p(\mathbf{x})$ from a large unlabelled set during training of classifiers?

One-class classification problems appear in practice when one of the classes is well sampled compared to other classes. The unbalance in sampling can be due to the measuring cost or the low frequency of occurrence. All multi-class classifiers have low performance in unbalanced problems. To overcome this we can focus on a description of the well sampled classes. The well sampled classes can be described by one-class classifiers since one-class classifiers do not suffer from unbalanced problems. They can be trained on examples from a single class only and if additional examples from other classes are available their performance improves further. This problem has been studied by us in part 1 and 2 of the thesis.

An alternative solution is to use standard multi-class classifiers and try to overcome the problem of unbalanced classes by knowledge from a given large unlabelled set. This framework was investigated in part 3. In general, multi-class classifiers perform badly for unbalanced problems. Therefore, one can modify the sampling of the objects to overcome the class unbalance. We can ask for additional labelled objects from the undersampled class or ask for the most informative objects from the undersampled class. If we minimise the number of label requests, this technique is called active learning. If we give a cost to retrieving a label of each unlabelled object, the active learning can be viewed a the sampling technique that achieves some classification error with the minimum label cost.

Alternatively, we can use a large unlabelled set to accurately estimate $p(\mathbf{x})$ and introduce this additional knowledge into the training of classifiers. This can be done in the semi-supervised framework.

10.1 Contribution

In part 1, we have discussed one-class classification models. The models were divided into two groups: statistically-based one-class classifiers, which are driven by the probability density of a target class and domain-based one-class classifiers, which are driven by the geometrical shape of the target class. The simplest difference between the two groups is that if we have several objects at the same position in the input space the statistically based one-class classifiers are influenced by these additional objects and the domain driven one-class classifiers treat these objects as one. Therefore, one-class classifiers which are domain-based are less sampling dependent. The statistically-based one-class classifiers require that data have to be sampled independently and identically to the true distribution while domain-based classifiers can achieve good performance when data is not sampled according to its probability density. Therefore, domain-based classifiers outperform statistically based classifiers in small sample size problems where it is hard to reliably estimate probability densities. On the other hand, when the target class is well sampled, then the probability density of the target class can be reliably estimated and consequently statistically based one-class classifiers might outperform domain-based classifiers in terms of the classification error.

We have proposed two domain-based one-class classifiers. The first classifier is based on the minimum volume enclosing ellipsoid (MVEE) algorithm. The problem of MVEE has been formulated as a conic optimisation problem. Comparing to other parametric methods, minimum volume enclosing box, minimum volume enclosing sphere and the single Gaussian one-class classifier, the MVEE has the smallest volume for most real-world problems for a given training set. This indicates a tight description of a target class. If we assume that an outlier class is uniformly distributed, the minimum volume descriptor indicates also the smallest error on the outlier class.

In addition, since the MVEE can be influenced by the presence of outlier objects in the training set we introduce an algorithm which computes the MVEE on the user specified fraction of data. The algorithm rejects a fraction of objects to estimate the MVEE. The MVEE is computed on the core set with the influence of rejected objects proportional to their slacks.

Finally when labelled outlier objects are available during training a third MVEE algorithm is proposed, which also uses outlier objects to estimate the ellipsoid with the minimum volume. However, such problems can be non-convex. The convexity of the problem depends on the number and the location of labelled outlier objects.

In high dimensional spaces the single Gaussian and the MVEE have similar performance since for normally distributed data all objects are on the surface of the estimated ellipsoids.

The second proposed one-class classifier is based on the minimum spanning tree algorithm, the minimum spanning tree data description (MST_DD). The basic elements of the proposed classifier are the edges of the graph. The distance from a test object to the target class is measured as the distance to the closest edge of the MST. The edges of the graph can be considered as additional training objects. Therefore MST_DD performs well in small sample size problems especially when data is distributed on a lower dimensional manifold. The MST_DD has no parameter to be set by the user, therefore is completely determined by the given data.

On the other hand, when the target class is normally distributed there are many redundant edges in the MST_DD. Therefore, we introduce an additional parameter which can be set to modify the complexity of the model.

In part 2, we focus on the model selection in the one-class classification problem. Since in general, examples of outlier objects are not available during training of one-class classifiers it is hard to select a model for a given dataset. To minimise the error on the target and

outlier classes, an assumption should be made about the distribution of outliers. One of the possibilities is to assume that the outlier data is uniformly distributed. In such a case, in addition to the minimisation of the error on the target class the volume of the classifier should be minimised. However, in high dimensional spaces a large amount of uniformly distributed outliers should be generated. This makes the algorithm impractical for such problems. We have proposed a similar model selection criterion to uniform outliers generation, although instead of minimising the total volume of the classifier we have proposed to minimise only the part of the classifier which is empty, i.e. without training objects. The empty region is approximated by the largest empty N-sphere that can be entirely found inside the classifier.

The proposed criterion, called $\mathcal{V}$-statistic, is based on the ratio of two volumes: the volume of the largest empty N-sphere inside the classifier, divided by the volume of the classifier. The $\mathcal{V}$-statistic measures how well the model fits the data. For a small value of the $\mathcal{V}$-statistic, the volume of classifier is large compared to the volume of the empty regions inside the classifier. A large classifier volume indicates a good generalisation performance on the target class and a small empty space indicates a good fit on the target class. There are not large empty regions in the classifier this means that the chance of accepting outlier objects is minimised.

The $\mathcal{V}$-statistic has been compared with other model selection methods based on uniform outlier generation and consistency model selection. In comparison with uniform outlier generation, the $\mathcal{V}$-statistic can be used in higher dimensional problems. However in high dimensions, volumes of the two N-spheres become similarly so that the $\mathcal{V}$-statistic approaches 1.

In part 3, we have investigated how to enhance the classification performance using a given set of unlabelled data. Since one-class classification problems arise when sampling of the classes is unbalanced, we try here to train multi-class classifiers using a set of unlabelled data to improve the performance of these classifiers. Two frameworks were investigated: active and semi-supervised learning.

In active learning, we proposed two sampling strategies called `vila` and `pdc`. The `vila` and `pdc` outperform standard active learning methods in problems where classes are multi-modal and exploration of input space is required. For problems where classes overlap it is possible to select a smaller fraction of labelled objects for which classifier has higher performance than the classifier trained on all labelled objects.

We have proposed a semi-supervised method which is based on the stability of soft-labels. The proposed method performs well when cluster structures can be related to classes. For continues distributions of $p(\mathbf{x})$ the improvement heavily depends on the given labelled samples.

In the thesis we have studied the problem of recognition and learning in the pattern recognition framework. Currently, in the field of pattern recognition mostly static problems are studied, i.e. the classifiers are trained once for a given dataset. There are little studies concerning problems changing in time.

As the main stream of pattern recognition focusses on two class classification, which is basically looking for a function that separates the classes, it is hard to justify the word "recognition" in the name of the field. Perhaps pattern discrimination is a more justifiable name.

10.2 Open questions

Although we have solved many problems there are plenty unsolved ones left in the topics we have studied. Also the algorithms we have presented rise many additional questions. Here is a list of some of them:

1. In the thesis we have solved problems which were represented in given vector spaces. The open question rises if this is optimal or at least reasonable representation for one-class classification problems and classification problems in general?

2. We have not studied the problem of feature selection. How to, based on a single class, select features or find easier to describe, more spherical, representation for a target class?

3. In general we use the Euclidian distance to compute relations between objects in $\mathbb{R}^N$. Is this a good measure? Are there better alternatives? Can we find good distance measures for a given problem?

4. The error is defined as the probability that a randomly selected object from the distribution is misclassified. However, in one-class classification we describe a single class against all other classes. Therefore, the proper estimation of the error should be close to one. Are there any alternatives for measuring the performance for one-class classifiers?

5. The domain-based one class classifiers are based on shapes or distances between objects. Is the error, as based on probabilities, a good measure to evaluate such classifiers?

6. Because the error is based on random sampling of objects, can we develop a similar measure for objects sampled in no random way, like in active learning?

7. We have discussed many active learning functions, however in practice one faces a problem of selecting a single one. How, based on few labelled objects, and perhaps some knowledge about a problem one should select a particular active learning function? What is a good criterion to do it?

8. Can we select automatically from a group of active learning functions, during the active learning process, which function should be chosen to select next query?

Appendices A

One-class classifiers

A.1 Volume of a single Gaussian data description with a given threshold

The equation of a single Gaussian data description with a given threshold is expressed as:

$$\frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} = \theta \tag{A.1}$$

The above equation can be rewritten to the standard equation of an ellipsoid:

$$(\mathbf{x}-\boldsymbol{\mu})^T - \frac{\Sigma^{-1}}{2\ln(\sqrt{(2\pi)^N\theta\det(\Sigma)})}(\mathbf{x}-\boldsymbol{\mu}) = 1 \tag{A.2}$$

Therefore we can use equation (3.2) with the matrix:

$$E = -\frac{\Sigma^{-1}}{2\ln(\sqrt{(2\pi)^N\theta\det(\Sigma)})} \tag{A.3}$$

to compute volume of a Gaussian with a threshold θ and a covariance matrix Σ.

A.2 Minimum volume enclosing ellipsoid

We start from the minimisation problem (3.11). The data has been mapped from $\mathbb{R}^N$ to $\mathbb{R}^{N+1}$ $E \to \tilde{M}$, $\mathbf{x}_i \to \tilde{\mathbf{x}}_i$ and we optimise a new ellipsoid centred at the origin.

$$\min_{\tilde{M}} \quad -\ln\det(\tilde{M}), \tag{A.4a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T\tilde{M}\tilde{\mathbf{x}}_i \leq 1, \quad \forall i = 1,\ldots,n, \tag{A.4b}$$

$$\tilde{M} \succ 0. \tag{A.4c}$$

To define an ellipsoid, the matrix $\tilde{M}$ has to be symmetric positive definite. We can force the positiveness of $\tilde{M}$ by decomposing it on $\tilde{M} = UU^T$. Therefore, $\tilde{M} \succ 0$ iff there exists a $N+1 \times N+1$ matrix U. The minimisation (A.4) is expressed as:

$$\min_{U} \quad -\ln\det(UU^T), \tag{A.5a}$$

$$\text{s.t.} \qquad \tilde{\mathbf{x}}_i^T UU^T \tilde{\mathbf{x}}_i \leq 1, \quad \forall i = 1, \ldots, n. \tag{A.5b}$$

The Lagrangian of the minimisation (A.5) equals:

$$L(U) = -\ln\det(UU^T) - \sum_{i=1}^{n} \alpha_i (1 - \tilde{\mathbf{x}}_i^T UU^T \tilde{\mathbf{x}}_i) \tag{A.6}$$

by introducing the Lagrangian multipliers $\alpha_i \geq 0$ and setting the partial derivative over the elements u of the matrix U to zero gives the constrain:

$$\begin{aligned} \frac{\partial L}{\partial u} &= -\frac{1}{\det(UU^T)} 2\det(UU^T)U(UU^T)^{-1} + 2U\sum_{i=1}^{n} \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T = 0 \\ (UU^T)^{-1} &= \sum_{i=1}^{n} \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T \end{aligned} \tag{A.7}$$

where we used the following properties of a determinant derivatives [Lütkepohl, 1996]:

$$\frac{\partial \det(U^T U)}{\partial u} = 2\det(U^T U)U(U^T U)^{-1} \tag{A.8a}$$

$$\frac{\partial \tilde{\mathbf{x}} \det(U^T U)\tilde{\mathbf{x}}}{\partial u} = 2U\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T \tag{A.8b}$$

To satisfy the Kuhn-Tucker conditions [Rustagi, 1994] of the optimality for nonlinear programming the solution has to satisfy:

$$\sum_{i=1}^{n} \alpha_i^* (1 - \tilde{\mathbf{x}}_i^T U^* U^{*T} \tilde{\mathbf{x}}_i) = 0 \tag{A.9}$$

The second term in the above sum can be rearranged under the trace as follows:

$$\tilde{\mathbf{x}}_i U^* U^{*T} \tilde{\mathbf{x}}_i = \text{trace}(\tilde{\mathbf{x}}_i^T U^* U^{*T} \tilde{\mathbf{x}}_i) = \text{trace}(U^* U^{*T} \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T) \tag{A.10}$$

Therefore, the second term in equation (A.9) can be simplified as:

$$\begin{aligned} &\sum_{i=1}^{n} \alpha_i^* \tilde{\mathbf{x}}_i^T U^* U^{*T} \tilde{\mathbf{x}}_i = \sum_{i=1}^{n} \text{trace}(U^* U^{*T} \alpha_i^* \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T) = \text{trace}(U^* U^{*T} \sum_{i=1}^{n} \alpha_i^* \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T) \\ &= \text{trace}\left(\left(\sum_{i=1}^{n} \alpha_i^* \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T \right)^{-1} \sum_{i=1}^{n} \alpha_i^* \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T \right) = \text{trace}(I) = N + 1 \end{aligned} \tag{A.11}$$

Consequently, equation (A.9) equals zero only when $\sum_{i=1}^{n} \alpha_i^* = N + 1$. Therefore, the dual formulation of minimisation (A.5), considering (A.7) can be written now as:

$$\max_{\alpha_i} \quad \ln\det\sum_{i=1}^{n}\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T, \tag{A.12a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n}\alpha_i = N+1, \tag{A.12b}$$

$$0 \le \alpha_i \le 1, \quad \forall i = 1,\dots,n. \tag{A.12c}$$

A.3 Robust estimation of minimum volume enclosing ellipsoid

We start from the minimisation problem (3.15). The data has been mapped from $\mathbb{R}^N$ to $\mathbb{R}^{N+1}$. We replace $\tilde{M}$ by UU^T to demand the positiveness of the matrix $\tilde{M}$. Then the robust estimation of the MVEE is formulated as:

$$\min_{U,\rho,\xi_i} \quad -\ln\det(UU^T) + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \nu\rho, \tag{A.13a}$$

$$\text{s.t.} \quad \tilde{\mathbf{x}}_i^T UU^T\tilde{\mathbf{x}}_i \le \rho + \xi_i, \quad \forall i = 1,\dots,n, \tag{A.13b}$$

$$\xi_i \ge 0,\ \rho \ge 0, \quad \forall i = 1,\dots,n. \tag{A.13c}$$

where $\nu \ge 0$ is a user specified parameter indicating the fraction of objects outside an ellipsoid. The Lagrangian of the minimisation (A.13) is:

$$\begin{aligned} L(U,\rho,\xi_i) = &-\ln\det(UU^T) + \frac{1}{n}\sum_{i}^{n}\xi_i + \nu\rho - \sum_{i=1}^{n}\alpha_i(\rho+\xi_i-\tilde{\mathbf{x}}_i^T UU^T\tilde{\mathbf{x}}_i) \\ &- \sum_{i=1}^{n}\beta_i\xi_i - \gamma\rho \end{aligned} \tag{A.14}$$

with introducing the Lagrangian multipliers $\alpha_i, \beta_i, \gamma \ge 0$. Setting partial derivatives over u, ξ_i, ρ to zero gives the constrain:

$$\begin{aligned} &\frac{\partial L}{\partial u} = 0, && (UU^T)^{-1} = \sum_{i=1}^{n}\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T, \\ &\frac{\partial L}{\partial \xi_i} = 0, && \beta_i = \frac{1}{n} - \alpha_i, \qquad \alpha_i \le \frac{1}{n} \\ &\frac{\partial L}{\partial \rho} = 0, && \sum_{i=1}^{n}\alpha_i = \nu - \gamma, \qquad \nu \ge \gamma. \end{aligned} \tag{A.15}$$

substituting (A.15) into (A.14) gives:

$$\begin{aligned} L(U,\rho,\xi_i) = &\ln\det\sum_{i=1}^{n}\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T + \frac{1}{n}\sum_{i=1}^{n}\xi_i + \nu\rho - \nu\rho + \nu\gamma - \sum_{i=1}^{n}\alpha_i\xi_i + N + 1 \\ &- \frac{1}{n}\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i\xi_i - \gamma\rho = \ln\det\sum_{i=1}^{n}\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T + N + 1 \end{aligned} \tag{A.16}$$

To satisfy the Kuhn-Tucker conditions [Rustagi, 1994] of the optimality for nonlinear programming the solution has to satisfy:

$$\begin{aligned} \alpha_i^*(\rho^* + \xi_i^* - \tilde{\mathbf{x}}_i U^* U^{*T}\tilde{\mathbf{x}}_i^T) = 0, & \\ \beta_i^*\xi_i^* = 0, \qquad (\frac{1}{n} - \alpha_i^*)\xi_i^* = 0, & \\ \gamma^*\rho^* = 0 & \end{aligned} \tag{A.17}$$

For a non-zero volume ellipsoid $\rho^* \neq 0$ therefor $\gamma^* = 0$ and consequently from (A.15) we have $\sum_{i=1}^{n}\alpha_i = \nu$. Therefore, the dual of (A.13) is:

$$\max_{\alpha_i} \quad \ln\det\sum_{i=1}^{n}\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T, \tag{A.18a}$$

$$\text{s.t.} \quad \sum_{i=1}^{n}\alpha_i = \nu, \tag{A.18b}$$

$$0 \leq \alpha_i \leq \frac{1}{n}, \quad \forall i = 1,\ldots,n. \tag{A.18c}$$

ρ^* and slack variables ξ_i^* can be computed as:

$$\begin{aligned} &\alpha_i^*(\rho^* + \xi_i^* - \tilde{\mathbf{x}}_i^T\tilde{M}^*\tilde{\mathbf{x}}_i) = 0, \quad \forall(\alpha_i^* = \frac{1}{n}), \\ &\text{where} \quad \rho^* = \tilde{\mathbf{x}}_i^T\tilde{M}^*\tilde{\mathbf{x}}_i - \xi_i^*, \quad \forall(0 \leq \alpha_i^* < \frac{1}{n}), \end{aligned} \tag{A.19a}$$

$$(\frac{1}{n} - \alpha_i^*)\xi_i^* = 0, \quad \forall(0 \leq \alpha_i^* < \frac{1}{n}). \tag{A.19b}$$

A.4 Estimation of the minimum volume enclosing ellipsoid in the presence of an outlier class

We start from the minimisation problem (3.20). The data has been mapped from $\mathbb{R}^N$ to $\mathbb{R}^{N+1}$. We replace $\tilde{M}$ by UU^T to demand the positiveness of the matrix $\tilde{M}$. Then the optimisation (3.20) can be written as:

$$\min_{U,\rho,\xi_i,\xi_j} \quad -\ln\det(UU^T) + \frac{1}{k}\sum_{i=1}^{k}\xi_i + \nu\rho, \tag{A.20a}$$

$$\text{s.t.} \quad \omega_i\tilde{\mathbf{x}}_i^T UU^T\tilde{\mathbf{x}}_i \leq \omega_i\rho + \xi_i, \quad \forall i = 1,\ldots,k, \tag{A.20b}$$

$$\xi_i, \rho \geq 0,\ \omega_i \in \{1,-1\}, \quad \forall i = 1,\ldots,k. \tag{A.20c}$$

The Lagrangian of the minimisation (A.20) is:

$$\begin{aligned} L(U,\rho,\xi_i) = &-\ln\det(UU^T) + \frac{1}{k}\sum_{i=1}^{k}\xi_i + \nu\rho - \sum_{i=1}^{k}\alpha_i(\omega_i\rho + \xi_i - \omega_i\tilde{\mathbf{x}}_i^T UU^T\tilde{\mathbf{x}}_i) \\ &- \sum_{i=1}^{k}\beta_i\xi_i - \gamma\rho \end{aligned} \tag{A.21}$$

with introducing the Lagrangian multipliers $\alpha_i, \beta_i, \gamma \geq 0$. Setting partial derivatives over u, ξ_i, ρ to zero gives the constrain:

$$\begin{aligned} \frac{\partial L}{\partial u} &= 0, \qquad (UU^T)^{-1} = \sum_{i=1}^{k}\omega_i\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T, \\ \frac{\partial L}{\partial \rho} &= 0, \qquad \sum_{i=1}^{k}\omega_i\alpha_i = \nu - \gamma, \\ \frac{\partial L}{\partial \xi_i} &= 0, \qquad \beta_i = \frac{1}{k} - \alpha_i \Rightarrow \alpha_i \leq \frac{1}{k}. \end{aligned} \tag{A.22}$$

substituting (A.22) into (A.21) gives:

$$\begin{aligned} L(U,\rho,\xi_i) = &\sum_{i=1}^{k}\omega_i\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T + \frac{1}{k}\sum_{i=1}^{k}\xi_i + \nu\rho + \gamma\rho - \nu\rho - \sum_{i=1}^{k}\alpha_i\xi_i + N\sum_{i=1}^{k}\omega_i \\ &- \frac{1}{k}\sum_{i=1}^{k}\xi_i + \sum_{i=1}^{k}\alpha_i\xi_i - \gamma\rho = \sum_{i=1}^{k}\omega_i\alpha_i\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T + N\sum_{i=1}^{k}\omega_i \end{aligned} \tag{A.23}$$

To satisfy the Kuhn-Tucker conditions of the optimality for nonlinear programming the solution has to satisfy:

$$\begin{aligned} \gamma^*\rho^* = 0, &\qquad \rho^* \neq 0\ , \gamma^* = 0,\ \Rightarrow\ \sum_{i=1}^{k}\omega_i\alpha_i^* = \nu, \\ \alpha_i^*(\omega_i\rho^* + \xi_i^* - \omega_i\tilde{\mathbf{x}}_i U^* U^{*T}\tilde{\mathbf{x}}_i^T) = 0, &\qquad \forall(\alpha_i^* = \frac{1}{k}), \\ \beta_i^*\xi_i^* = 0, &\qquad (\frac{1}{k} - \alpha_i^*)\xi_i^* = 0, \qquad \forall(0 \leq \alpha_i^* \leq \frac{1}{k}). \end{aligned} \tag{A.24}$$

Therefore, the dual of the optimisation (A.20) is:

$$\max_{\alpha_i} \quad \ln\det \sum_{i=1}^{k} \omega_i \alpha_i \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T, \tag{A.25a}$$

$$\text{s.t.} \quad \sum_{i=1}^{k} \omega_i \alpha_i = \nu, \tag{A.25b}$$

$$0 \leq \alpha_i \leq \frac{1}{k}. \tag{A.25c}$$

ξ_i^* and ρ^* can be computed as:

$$\alpha_i^*(\omega_i \rho^* + \xi_i^* - \omega_i \tilde{\mathbf{x}}_i U^* U^{*T} \tilde{\mathbf{x}}_i^T) = 0, \qquad \forall(\alpha_i^* = \frac{1}{k}), \tag{A.26a}$$

$$(\frac{1}{k} - \alpha_i^*)\xi_i^* = 0, \qquad \forall(0 \leq \alpha_i^* \leq \frac{1}{k}). \tag{A.26b}$$

Appendices B

Volume based model selection in one-class classification

B.1 The volume of a spherical cap

The spherical cap is a part of an N-sphere as shown in figure B.1. When two N-spheres $\mathcal{S}(\mathbf{A}_1, R_1)$ and $\mathcal{S}(\mathbf{A}_2, R_2)$ intersect, a height h_c and a radius r_c of the two cups can be derived simply from Pythagoras equations as:

$$h_{c_1} = R_1 - \frac{\|\mathbf{A}_1 - \mathbf{A}_2\|^2 - R_2^2 + R_1^2}{2\|\mathbf{A}_1 - \mathbf{A}_2\|} \qquad h_{c_2} = R_2 - \frac{\|\mathbf{A}_1 - \mathbf{A}_2\|^2 + R_2^2 - R_1^2}{2\|\mathbf{A}_1 - \mathbf{A}_2\|}$$

$$r_{c_1} = \sqrt{R_1^2 - (R_1 - h_{c_1})^2} \qquad = \qquad r_{c_2} = \sqrt{R_2^2 - (R_2 - h_{c_2})^2}$$

The volume of a single cap can be computed by integrating the volumes of $N-1$ dimensional spheres from the radius r_c till 0 over different value of a height h_c.

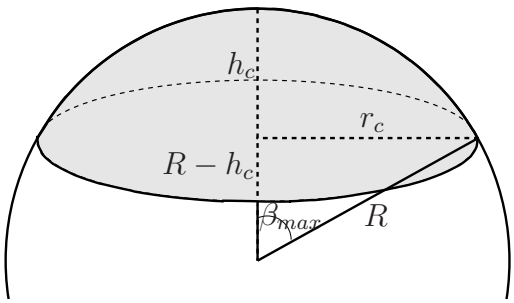


Figure B.1: Spherical cap.

$$V_{cap} = \frac{2\pi^{(N-1)/2}}{\Gamma((N-1)/2+1)} \int_0^{h_c} r_c^{N-1}(h_c) dh_c$$

From figure B.1 it can be seen that:

$$r_c^2 + (R - h_c)^2 = R^2,$$
$$r_c = R\sin(\beta_{max}).$$

Substituting those equations gives:

$$V_{cap} = \frac{2\pi^{(N-1)/2} R^{N-1}}{\Gamma((N-1)/2+1)} \int_0^{\beta_{max}} \sin^{N-1}(\beta) d\beta \quad \text{(B.1a)}$$

$$\beta_{max} = \arcsin(\sqrt{(2R - h_c)(h_c/R^2)}) \quad \text{(B.1b)}$$

The integral $\int \sin^{N-1}(\beta) d\beta$ can be handled by recursion [Bronshtein et al., 1997, § 8].

$$\int \sin^{N-1}(\beta) d\beta = -\frac{\sin^{N-2}\beta \cos\beta}{N-1} + \frac{N-2}{N} \int \sin^{N-3}\beta d\beta \quad \text{(B.2)}$$

B.2 Overlapping *N*-spheres problem

To check if two N-spheres $\mathcal{S}(\mathbf{A}_i, R_i)$, $i = 1, 2$ overlap it is sufficient to check that

$$\|A_1 - A_2\| \leq R_1 + R_2 \tag{B.3}$$

To check if $k > 2$ N-spheres overlap in such way that they all have a common region Y $Y = \{\mathcal{S}(\mathbf{A}_1, R_1) \cap \mathcal{S}(\mathbf{A}_2, R_2) \cap \ldots \cap \mathcal{S}(\mathbf{A}_k, R_k) : |Y| > 1\}$ it is sufficient to find a single vector $\mathbf{y} \in Y$ which is inside all N-spheres:

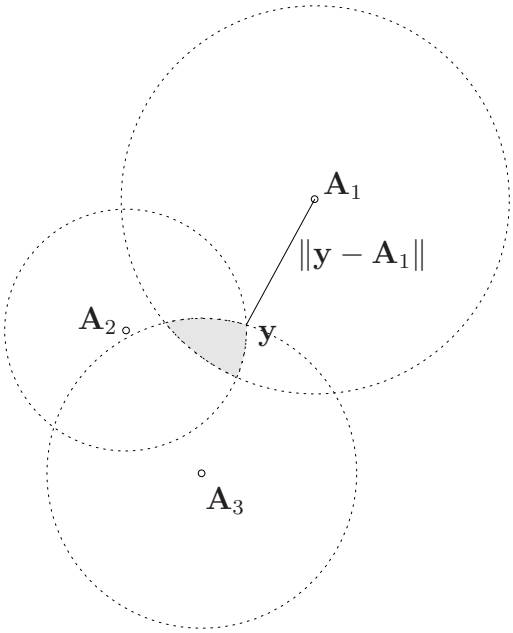


Figure B.2: Three overlapping sphere problem

$$\begin{aligned} &\mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{A}_i + \mathbf{A}_i^T\mathbf{A}_i < R_i^2, \\ &i = 1, 2 \ldots, k. \end{aligned} \tag{B.4}$$

To find a single vector $\mathbf{y}$ we can reformulate the set of k inequalities (B.4) into quadratic programming problem:

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{A}_1 + \mathbf{A}_1^T\mathbf{A}_1, \\ \text{st.} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{A}_i + \mathbf{A}_i^T\mathbf{A}_i - R_i^2 < 0, \\ & i = 2, \ldots, k. \end{aligned} \tag{B.5}$$

Figure B.2 provides graphical illustration. If $\exists \mathbf{y}$ and $\|\mathbf{y} - A_1\| < R_1$ then k N-spheres have a common region.

B.3 Derivation of a centre of a *N*-sphere from $N + 1$ objects

Consider $N+1$ affinely independent points $\mathbf{x}_i$, $i = 1, 2, \ldots, N+1$ in a N-dimensional space, located on the surface of the N-sphere $\mathcal{S}(\mathbf{a}, r)$. As the points $\mathbf{x}_i$ are equidistant from the centre $\mathbf{a}$, one can write:

$$||\mathbf{x}_i - \mathbf{a}||^2 = ||\mathbf{x}_j - \mathbf{a}||^2, \qquad \forall_{i,j=1,\ldots,N+1}. \tag{B.6}$$

The centre of the N-sphere can be determined by solving the above system of equations. By straightforward calculations, this simplifies to:

$$(\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{a} = \frac{1}{2}(\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_j^T\mathbf{x}_j), \qquad \forall_{i,j=1,\ldots,N+1}. \tag{B.7}$$

Multiplying both sides by the vector $(\mathbf{x}_i - \mathbf{x}_j)$, we get:

$$(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{a} = \frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_j^T\mathbf{x}_j), \qquad \forall_{i,j=1,\ldots,N+1}. \tag{B.8}$$

Neglecting situations where $i = j$, which give zeros, we have $\binom{N+1}{2}$ equations. We can rewrite this set of equations in the more specific way. First, we replace $\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_j^T\mathbf{x}_j$ by $(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i + \mathbf{x}_j)$. Secondly we define X_- as a $N \times \binom{N+1}{2}$ matrix with elements $\mathbf{x}_i - \mathbf{x}_j$ $\forall i, j = 1, \ldots, N+1,\ i \neq j$ and X_+ a $N \times \binom{N+1}{2}$ matrix with elements $\mathbf{x}_i + \mathbf{x}_j$ $\forall i, j = 1, \ldots, N+1,\ i \neq j$. Therefore equations (B.8) can be written now as a single equation:

$$X_- X_-^T \mathbf{a} = \frac{1}{2} X_- (\mathbf{1}^T X_-^T X_+)^T. \tag{B.9}$$

where $\mathbf{1}$ is the $\binom{N+1}{2} \times 1$ vector of ones. Note that non-singularity of a $N \times N$ matrix $X_- X_-^T$ is guaranteed by the non-collinearity of the points $\mathbf{x}_i$. Therefore, the sought centre $\mathbf{a}$ can be computed by:

$$\mathbf{a} = \frac{1}{2} (X_- X_-^T)^{-1} X_- (\mathbf{1}^T X_-^T X_+)^T. \tag{B.10}$$

B.4 Check whether a N-sphere is inside a union of N-spheres

We would like to find the vector $\mathbf{y}$ that is inside $\mathcal{S}(\mathbf{z}, \rho)$ and it is outside the classifier defined by a union of k N-spheres $\bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$.

$$\begin{aligned} \|\mathbf{y} - \mathbf{z}\|^2 &\leq \rho^2 \\ \|\mathbf{y} - \mathbf{A}_i\|^2 &\geq R_i^2, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{B.11}$$

by rearranging above inequalities:

$$\begin{aligned} \mathbf{y}^T\mathbf{y} &\leq 2\mathbf{y}^T\mathbf{z} + \rho^2 - \mathbf{z}^T\mathbf{z} \\ \mathbf{y}^T\mathbf{y} &\geq 2\mathbf{y}^T\mathbf{A}_i + R_i^2 - \mathbf{A}_i^T\mathbf{A}_i, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{B.12}$$

therefore $\mathbf{y}^T\mathbf{y}$ can be bounded by:

$$2\mathbf{y}^T\mathbf{A}_i + R_i^2 - \mathbf{A}_i^T\mathbf{A}_i \leq \mathbf{y}^T\mathbf{y} \leq 2\mathbf{y}^T\mathbf{z} + \rho^2 - \mathbf{z}^T\mathbf{z}, \quad \forall_{i=1,\ldots,k} \tag{B.13}$$

We drop $\mathbf{y}^T\mathbf{y}$ and rearrange above k inequalities. Together with the first inequality from (B.12) our problem can be now written as:

$$\begin{aligned} \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{z} &\leq \rho^2 - \mathbf{z}^T\mathbf{z} \\ 2\mathbf{y}^T(\mathbf{A}_i - \mathbf{z}) &\leq \rho^2 + \mathbf{A}_i^T\mathbf{A}_i - R_i^2 - \mathbf{z}^T\mathbf{z}, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{B.14}$$

This set of inequalities can be replaced by the following quadratic minimisation problem:

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\mathbf{z} \\ \text{s.t.} \quad & 2\mathbf{y}^T(\mathbf{A}_i - \mathbf{z}) \leq \rho^2 + \mathbf{A}_i^T\mathbf{A}_i - R_i^2 - \mathbf{z}^T\mathbf{z}, \quad \forall_{i=1,\ldots,k} \end{aligned} \tag{B.15}$$

If $\mathbf{y}$ satisfy first inequality in (B.14) a part or entire a N-sphere $\mathcal{S}(\mathbf{z}, \rho)$ is outside $\bigcup_{i=1}^{k} \mathcal{S}(\mathbf{A}_i, R_i)$.

Summary Learning to recognise

The thesis treats classification problems which are undersampled or where there exist an unbalance between classes in the sampling. The thesis is divided into three parts. The first two parts treat the problem of one-class classification. In the one-class classification problem, it is assumed that only examples of one of the classes, the target class, are available. The fact that no (or almost no) examples of other classes are available makes the one-class classification an example of an extremely unbalance problem. Therefore, such problem can not be described accurately by existing multi-class classifiers. However, a need to solve such classification rises from many theoretical and practical problems, e.g. the concept learning, machine fault detection and face recognition.

In the third part of the thesis, we treat classification problems which are undersampled but not necessary unbalanced. In such problems, additional examples or additional knowledge about data available during training significantly improves classification performance. We investigate two types of enhancement of a small training set with additional knowledge from a large unlabelled data set: active learning and semi-supervised sampling.

Chapter 1 introduces a reader to the problem of one-class classification. The chapter compares the one-class classification approach to the multi-class classification approach explaining advantages and disadvantages of both approaches. Next, a short introduction to one-class classifiers is given, followed by a description of performance measures of classifiers in the one-class classification problem.

Chapter 2 gives the general introduction to one class classification models. The one-class classifiers are divided into two groups: statistical-based one-class classifiers and domain-based one-class classifiers. The statistical-based classifiers are classifiers based on the probability density function of a target class. The domain-based one-class classifiers are determined by the shape of the target class and not by frequency of objects.

Chapter 3 introduces a new domain-based one-class classifier. The classifier is based on the **m**inimum **v**olume **e**nclosing **e**llipsoid (MVEE) algorithm. The algorithm is formulated as a conic programming problem. We give a prime and dual for three conic problems based on the MVEE. The first ellipsoid is computed as an ellipsoid with minimum volume, enclosing all target objects. The second ellipsoid is more robust version of the previous classifier and it is computed as the minimum volume ellipsoid with a user specified fraction of target objects outside the ellipsoid. The third ellipsoid is computed in situations where both target and outlier objects are available during training. Parameters of the ellipsoid are determined by minimising distances of misclassified objects to the surface of the ellipsoid.

Chapter 4 introduces a new domain-based one-class classifier. The classifier is based on a graph description of the target class. In particular, the minimum spanning three is used to describe

the target class. The distance from a test object to the target class is measured as a distance from the test object to the closed edge of the graph. An additional complexity parameter is introduced to adapt the complexity of the graph model to the complexity of the target class.

Chapter 5 gives an introduction to the problem of model selection in one-class classification problems. As it is assumed that only target objects are available during training of one-class classifiers it is especially difficult to select a model with small error on the target and outlier class. Model selections based on the uniform distributed outlier objects and on the consistency of a one-class classifier are described.

Chapter 6. In this chapter a new model selection criterion for one-class classifiers is introduced. The criterion is based on the ratio of two volumes: the volume of the largest empty N-sphere that can be found inside a one-class classifier divided by the volume of the one-class classifier. To compute the ratio of two volumes several subproblems have been solved. In section 6.1.1 we present a formula to compute a tight approximation of the volume of one-class classifiers consisting of several intersecting N-spheres such as k-means, k-centres and self-organising maps. The proposed approach can tightly approximate the volume of a given classifier in any number of dimensions. In the same section, we derive a formula to compute the volume of a spherical cap in arbitrary number of dimensions and present a method to check whether more than two N-spheres have a common region. Next, in section 6.1.2 we propose an algorithm to find the largest empty N-sphere in one-class classifiers consisting of N-spheres. Here we propose a method to check whether a N-sphere is entirely inside a set of intersecting N-spheres. Section 6.1.3 presents an explanation why the presented algorithm does not work for kernel based one-class classifiers such as SVDD and *oc*-SVM. Finally, an approximate largest N-sphere search algorithm is presented in section 6.1.4 that is applicable to any one-class classifiers.

Chapter 7 first gives a general introduction to the problem of active learning. Next, we introduce two new active learning methods. The first method (`vila`) is based on **v**ariation **i**n **l**abel **a**ssignments of unlabelled data by a set of classifiers. Each classifier, from the set, is trained on a given training set enlarged with a single unlabelled object assigned to one of the classes from the training set. The unlabelled object is selected which gives the largest variation of classification labels of unlabelled objects.

The second active learning function is based on **p**ositive **d**ensity **c**orrection (`pdc`). The unlabelled object is selected to be added to the training set for which the difference between the density of unlabelled set and the training set has the largest positive value.

Chapter 8. In active learning usually a single query is selected in a trial. This chapter points out to problems where multiple queries need to be selected in a single trial. As the queries selected by most active learning algorithms are similar additional diversification algorithm should be introduced.

Chapter 9 describes a problem of semi-supervised learning, where one tries to link conditional class probability densities, estimated on a small training set, with a density of large unlabelled data. The goal is to improve the performance of classifiers by this additional information. We proposed a semi-supervised learning algorithm based on stability of soft labels. The algorithm is applicable to any density based classifiers.

Samenvatting Leren om te herkennen

Het proefschrift behandelt klassificatie problemen die onvoldoende bemonsterd zijn of waarvan de bemonstering van de klassen erg onevenwichtig is. Het proefschrift is verdeeld in drie delen. De eerste twee delen behandelen het probleem van één klasse klassificatie. In het één-klasse klassificatie probleem wordt aangenomen dat er alleen voorbeelden van één van de klassen, de 'doel' klasse, beschikbaar zijn. Het feit dat er (vrijwel) geen voorbeelden van de andere klassen aanwezig zijn, maakt het één-klasse klassificatie probleem een voorbeeld van een extreem onevenwichtig bemonsterd probleem. Hierdoor kan zo'n probleem niet goed beschreven worden door bestaande veel-klasse klassificatoren. Echter, er bestaat de behoefte op zulke klassificatie problemen op te lossen in vele theoretische en praktische problemen, zoals het leren van concepten, machine diagnostiek en gezichtsherkenning.

In het derde gedeelde van het proefschrift behandelen we problemen die onvoldoende bemonsterd zijn door het toevoegen van extra objecten gedurende de training van multi-klasse klassificatoren. De prestaties van de klassifiers verbeteren significant. We onderzoeken twee methoden om met extra kennis een kleine train set te verbeteren door gebruik te maken van een grote ongelabelde data set: 'actief leren' en 'semi-supervised leren'.

Hoofdstuk 1 introduceert de lezer in het probleem van de één-klasse klassificatie. Het hoofdstuk vergelijkt de één-klasse klassificatie benadering met de veel-klasse klassificatie benadering en laat de voor- en nadelen van beide benaderingen zien. Vervolgens wordt een korte introductie over één-klasse klassificatoren geven, gevolgd door een beschrijving van prestatie-maten voor klassificatoren in het één-klasse klassificatie probleem.

Hoofdstuk 2 geeft een algemene introductie in één-klasse klassificatie modellen. De één klasse klassificatoren worden verdeeld in twee groepen: de statistisch gebaseerde klassificatoren en de domein-gebaseerde klassificatoren. De statistisch gebaseerde klassifiers zijn gebaseerd op de waarschijnlijkheids verdelingsfunctie van de doel klasse. De domein-gebaseerde klassificatoren worden vastgesteld door de vorm van de doel klasse.

Hoofdstuk 3 introduceert een nieuwe domein-gebaseerde één-klasse klassificator. De klassificator is gebaseerd op het minimum-volume ellipsoide (Minimum Volume Enclosing Ellipsoid, MVEE) algorithme. Dit algorithme is geformuleerd as een conisch programmeer-probleem. We presenteren de primaal en duaal voor drie conisch programmeer probleem voor de MVEE. De eerste ellipsoide is de ellipsoide met een minimum volume die alle doel objecten insluit. De tweede ellipsoide is een robustere versie van de eerste classifier en wordt bepaald door een minimum volume waarbij een bepaalde, door de gebruiker gespecificeerde, fractie van objecten buiten de ellipsoide valt. De derde ellipsoide wordt bepaald ???

Hoofdstuk 4 introduceert een nieuwe domein-gebaseerde één-klasse klassificator. De klassificator is gebaseerd op een graaf-beschrijving van de doel klasse. De afstand tussen een

test object en de doel klasse wordt gedefinieerd als de afstand van dit test object naar de dichtsbijzijnde edge 'lijnstuk' (??) in de graaf. Een additionele complexiteits parameter is geintroduceerd om de graaf aan de complexiteit van de doel klasse aan te passen.

Hoofdstuk 5 geeft een introductie in het probleem van de model selectie in één klasse klassificatie problemen. Omdat wordt aangenomen dat alleen objecten uit de doel klasse beschikbaar zijn gedurende de training van de klassificatoren, is het moeilijk een model te selecteren dat een kleine fout op zowel de doel als de uitbijter klasse. De model selectie gebaseerd op een uniforme verdeling van uitbijter objecten en de consequentheid van een één-klasse klassificator wordt beschreven.

Hoofdstuk 6. In dit hoofdstuk wordt een nieuw model selectie criterium voor één-klasse klassificatoren voorgesteld. Het criterium is gebaseerd op de verhouding tussen twee volumes: het volume van de grootste lege bal dat binnen een één-klasse klassificator past en het volume van de één-klasse klassificator. Om de verhouding tussen de twee volumes te berekenen moeten verschillende subproblemen opgelost worden. In sectie 7.1.1 presenteren we een formule om een dichte benadering van het volume te schatten van één-klasse klassificatoren die uit een verzameling van elkaar snijdende bollen bestaan, zoals k-means, k-centres en Self-organising maps. De voorgestelde aanpak kan het volume van een gegeven klassificator dicht benaderen in elke dimensionaliteit. In dezelfde sectie leiden we een formule af om het volume van een spherische kap in een willekeurig aantal dimensies te berekenen en presenteren we een methode om te testen of meer dan twee bollen een gemeenschappelijk gebied innemen. Vervolgens in sectie 7.1.2 stellen we een algorithme voor om de grootste lege bal te vinden in één-klasse klassificatoren die bestaan uit elkaar snijdende bollen. Hier presenteren we een methode om te testen of een bol geheel binnen een set van snijdende bollen valt. Sectie 7.1.3 geeft een uitleg waarom deze methode niet werkt voor kernel-gebaseerde één-klasse klassificatoren zoals de SVDD en de oc-SVM. Tenslotte wordt in sectie 7.1.4 een benaderend zoek algorithme voor de grootste bal gepresenteerd dat op all één-klasse klassificatoren kan worden toegepast.

Hoofdstuk 7 geeft eerste een introductie over het probleem van 'actief leren'. Vervolgens introduceren we twee nieuwe methodes. De eerste methode (`vila`) is gebaseerd op de variatie in de label toewijzingen aan ongelabelde objected door een set van klassificatoren. Elke klassificator uit de set wordt getraind op een gegeven train set die is uitgebreid met één enkel ongelabeld object dat aan één van de klassen wordt toegewezen. Het object dat de grootste variatie van klassificatie labels voor de ongelabelde objecten geeft, wordt geselecteerd.

De tweede 'actief leer' methode is gebaseerd op positieve dichtheids correctie. Hier het object met het grootste positieve verschil tussen dichtheid van de ongelabelde set en de training set wordt gekozen.

Hoofdstuk 8. In actief leren wordt gewoonlijk één enkel object per experiment geselecteerd. Dit hoofdstuk laat problemen zien waarbij meerdere objecten moeten worden geselecteerd per experiment. Omdat de meeste active learning algoritmes dezelfde objecten selecteren moet er een extra diversificatie algorithme worden geintroduceerd.

Hoofdstuk 9 beschrijft het probleem van actief leren, waarin men de conditionele klasse dichtheden, geschat op een kleine training set, probeert te verbinden met de dichtheid van een grote

set ongelabelde data. Het doel is om de prestatie van klassificatoren met deze extra informatie te verbeteren. We stellen een semi-supervised algorithme voor dat is gebaseerd op de stabiliteit van soft labels. Dit algorithme is toepasbaar op alle dichtheidsgebaseerde klassifiers.

Curriculum vitae

Piotr Juszczak was born in Oława in Poland on November 11, 1976. In 1996 he finished the Chemical Technical Secondary School in Wrocław and went to study Physics at the Wrocław University of Technology in Poland. During his study he visited the Department of Electronic Engineering at the National University of Ireland, where he spent two years. In 2001 he obtained a Master degree in Physics. The subject of his Master thesis was "Automatic recognition of arrhythmia from a ECG signal". The research was conducted at the Department of Medical Physics under the supervision of Dr. Z. Moroń.

In 2002 he started his Ph.D. research in pattern recognition at the Delft University of Technology. The research was conducted under supervision of Dr.Ir. R.P.W Duin.

In 2006 he became a research associate at the Institute for Mathematical Sciences, Imperial Collage London.

Acknowledgements

For me, being a PhD student was a constant source of ups and downs, probably like for every PhD student. There were times when I would have liked to throw this thesis from the highest mountain in the Netherlands and times when I was very excited about my work. Finally, I made it, and this would not have been possible without the support and inspiration I have received from many people.

Bob Duin guided me through my whole PhD. He was always ready to give advice when I needed it and at the same time gave me the freedom and encouraged me to go my own way. It is amazing how much you can learn from a single person during such a short time. Whenever I came up with new ideas or results he shared my enthusiasm, always egging me further by asking exactly the right questions. Also, I truly enjoyed our philosophical discussions and escapades to jazz and classical concerts.

I am grateful to the people of the Neuro group for creating a truly joyful and supportive group of scientists to work with. They are: Bob Duin, Marina Skurichina, Dick de Ridder, David Tax, Elżbieta Pękalska, Pavel Paclík, Carmen Lai, Serguey Verzakov, Thomas Landgrebe and Artsiom Harol. Doing a PhD is one thing, but doing it in the Neuro group in such a stimulating environment, among friends, is another thing.

Most of all I am indebted to David Tax, without whom this thesis never would have been completed. In countless discussions he explained to me his view on one-class classifiers, patiently answered my questions, and gave valuable hints and suggestions for my work.

I would like to thank the members of the PH group at Delft University (which unfortunately no longer exist) for such a warm welcome during my first two years in that group as a PhD student. My special thanks to Micheal van Ginkel, Dick de Ridder, Cris Luengo Hendriks and Bernd Rieger for creating the wonderful atmosphere in the group.

I spent the second half of my PhD in the ICT group. I am grateful to the members of this group for their hospitality. My special thanks to Marcel Reinders for being a wonderful and demanding promotor and for pointing out several errors in an early draft of this thesis.

My thanks to the Polish community of former or present PhD students in Delft: Elżbieta and Andrzej Pękalska, Iwona and Hans Kordal, Ania and Jacek Wojdeł, Mirka and Leszek Góra, Ania and Wojtek Stec, Elwira and Adrian Bohdanowicz, Ewelina and Michał Sobera and almost Polish Gian Luca di Nola for enjoyable volleyball matches and organisation of countless parties.

Finally, I would like to thank my family, especially my parents who have told me everything I need to know in life.

Bibliography

[Abe and Mamitsuka, 1998] Abe, N. and Mamitsuka, H. (1998). Query learning strategies using boosting and bagging. In *Proceedings of the 15th International Conference on Machine Learning*, pages 1–9. Referred to on pages: 94

[Angluin, 1988] Angluin, D. (1988). Queries and concept learning. *Machine Learning*, 2(4):319–342. Referred to on pages: 92, 96

[Argamon-Engelson and Dagan, 1999] Argamon-Engelson, S. and Dagan, I. (1999). Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360. Referred to on pages: 94

[Baker et al., 1999] Baker, D., Hofmann, T., McCallum, A., and Yang, Y. (1999). A hierarchical probabilistic model for novelty detection in text. Technical report, Just Research. Referred to on pages: 18

[Baldi and Hornik, 1989] Baldi, P. and Hornik, K. (1989). Neural networks and principal component analysis: learning from examples without local minima. *Neural networks*, 2:53–58. Referred to on pages: 20

[Baram et al., 2004] Baram, Y., El-Yaniv, R., and Luz, K. (2004). Online choice of active learning algorithms. *Journal of Machine Learning Research*, 5:255–291. Referred to on pages: 92, 105

[Barla et al., 2002] Barla, A., Odone, F., and Verri, V. (2002). Hausdorff kernel for 3d object acquisition and detection. In *Proceedings of the European Conference on Computer Vision, LNCS 2353*, page 20. Referred to on pages: 23

[Barnes, 1982] Barnes, E. (1982). An algorithm for separating patterns by ellipsoids. *IBM Journal of Research Development*, 26(6):759–764. Referred to on pages: 25

[Barnett and Lewis, 1994] Barnett, V. and Lewis, T. (1994). *Outliers in Statistical data*. Wiley, New York, 3 edition. Referred to on pages: 5

[Ben-David and Lindenbaum, 1997] Ben-David, S. and Lindenbaum, M. (1997). Learning distributions by their density levels: A paradigm for learning without a teacher. *Journal of Computer and System Sciences*, 55(1):171–182. Referred to on pages: 17

[Beyer et al., 1999] Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is 'nearest neighbor' meaningful? In *Lecture Notes in Computer Science*, volume 540, pages 217–235. Referred to on pages: 20

[Bishop, 1994] Bishop, C. (1994). Mixture density networks. Technical Report NCRG 4288, Neural computation research group, Aston University, Birmingham. Referred to on pages: 18

[Bishop, 1995] Bishop, C. (1995). *Neural networks for pattern recognition*. Oxford University Press, Walton Street, Oxford OX2 6DP. Referred to on pages: 15, 17, 18, 19, 57, 121

[Blum et al., 2004] Blum, A., Lafferty, J., Rwebangira, M. R., and Reddy, R. (2004). Semi-supervised learning using randomised mincuts. In *Proceedings of the 21st International Conference on Machine Learning*. Referred to on pages: 122, 123

[Bourlard and Kamp, 1988] Bourlard, H. and Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59:291–294. Referred to on pages: 20

[Boyd and Vandenberghe, 2003] Boyd, S. and Vandenberghe, L. (2003). *Convex Optimization*. Cambridge University Press. Referred to on pages: 26

[Bradley, 1997] Bradley, A. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159. Referred to on pages: 7

[Brinker, 2003] Brinker, K. (2003). Incorporating diversity in active learning with support vector machine. In *Proceedings of the 20th International Conference on Machine Learning*, pages 59–66, Menlo Park, California. AAAI Press. Referred to on pages: 95, 112, 113

[Bronshtein et al., 1997] Bronshtein, I. N., Semendyayev, K. A., and Hirsch, K. A. (1997). *Handbook of Mathematics*. Springer-Verlag Telos. Referred to on pages: 141

[Campbell and Bennett, 2000] Campbell, C. and Bennett, K. P. (2000). A linear programming approach to novelty detection. In *Neural Information Processing Systems*, pages 395–401. Referred to on pages: 15, 23

[Chapelle, 2004] Chapelle, O. (2004). Active learning for parzen window classifier. Technical report, Max Planc institute for Biological Cibernetics. Referred to on pages: 105

[Chapelle et al., 2002] Chapelle, O., Weston, J., and Schöelkopf, B. (2002). Cluster kernels for semi-supervised learning. In *Advances Neural Information Processing Systems*, volume 15, pages 585–592. Referred to on pages: 122, 123

[Chow, 1970] Chow, C. (1970). On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, IT-16(1):41–46. Referred to on pages: 15, 17

[Cohen et al., 2004] Cohen, I., Cozman, F., Sebe, N., Cirelo, M., and Huang, T. (2004). Semi-supervised learning of classifiers: theory, algorithms, and their application to human-computer interaction. *PAMI*, 26(12):1553–1566. Referred to on pages: 128

[Cohn et al., 1995] Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1995). Active learning with statistical models. In Tesauro, G., Touretzky, D., and Leen, T., editors, *Advances in Neural Information Processing Systems*, volume 7, pages 705–712. MIT Press. Referred to on pages: 92

[Coleman and Li, 1996] Coleman, T. and Li, Y. (1996). An interior, trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6:418–445. Referred to on pages: 65

[Cormen et al., 1990] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press, Cambridge, MA. Referred to on pages: 42, 43

[Cox and Cox, 1994] Cox, T. F. and Cox, M. A. A. (1994). *Multidimensional scaling*. Chapman & Hall. Referred to on pages: 44

[Cramér, 1946] Cramér, H. (1946). *Mathematical methods of statistics*. Princeton University Press, Prinston, N.J. Referred to on pages: 72

[Dasgupta et al., 2005] Dasgupta, S., Kalai, A., and Monteleoni, C. (2005). Analysis of perceptron-based active learning. In *Conference on Learning Theory*. Referred to on pages: 95

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal Roy. Stat. Soc. B*, 39:1–38. Referred to on pages: 19

[Duda et al., 2001] Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern classification*. John Wiley & Sons, second edition. Referred to on pages: 2, 121, 123, 124, 125

[Duin, 1976] Duin, R. (1976). On the choice of the smoothing parameters for Parzen estimators of probability density functions. *IEEE Transactions on Computers*, C-25(11):1175–1179. Referred to on pages: 18, 54, 97, 115, 126

[Duin, 1999] Duin, R. P. W. (1999). Compactness and complexity of pattern recognition problems. In *Internat. Symposium on Pattern Recognition 'In Memoriam Pierre Devijver'*, pages 124–128, Royal Military Academy, Brussels. Referred to on pages: 25, 55

[Duin et al., 2004] Duin, R. P. W., Pekalska, E., and Tax, D. M. J. (2004). The characterization of classification problems by classifier disagreements. In Kittler, J., Petrou, M., and Nixon, M., editors, *17th International Conference on Pattern Recognition*, volume 2, pages 140–143. IEEE Computer Society, Los Alamitos, CA. Referred to on pages: 44

[Fine et al., 2002] Fine, S., Gilad-Bachrach, R., and Shamir, E. (2002). Query by committee, linear separation and random walks. *Theoretical Computer Science*, 284(1):25–51. Referred to on pages: 94

[Freund et al., 1997] Freund, Y., Seung, H. S., Shamir, E., and Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2–3):133–168. Referred to on pages: 92, 95, 108

[Fukunaga, 1972] Fukunaga, K. (1972). *Introduction to statistical pattern recognition*. Academic Press. Referred to on pages: 126

[Gower, 1986] Gower, J. C. (1986). Metric and euclidean properties of dissimilarity coefficients. *Journal of Classification*, 3:5–48. Referred to on pages: 40, 110

[Graham and Hell, 1985] Graham, R. L. and Hell, P. (1985). On the history of the minimum spanning tree problem. *Annals of History of Computing*, 7(1):43–57. Referred to on pages: 41

[Gritzmann and Klee, 1993] Gritzmann, P. and Klee, V. (1993). Computational complexity of inner and outer j-radii of polytopes in finite-dimensional normed spaces. *Math. Program.*, 59(2):163–213. Referred to on pages: 63

[Grötschel et al., 1998] Grötschel, M., Lovasz, L., and Schrijver, A. (1998). *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag. Referred to on pages: 27

[Harmeling et al., 2005] Harmeling, S., Dornhege, G., Tax, D. M. J., Meinecke, F., and Mueller, K. R. (2005). From outliers to prototypes: ordering data. *Neurocomputing*. Referred to on pages: 19, 20

[Harris and Stocker, 1998] Harris, J. W. and Stocker, H. (1998). *Handbook of mathematics and computational science*. Springer-Verlag, New York. Referred to on pages: 59

[Hart, 1968] Hart, P. (1968). The condensed nearest neighbor rule. *IEEE Transactions on Information theory*, 14:505–516. Referred to on pages: 19

[Hasenjäger et al., 1999] Hasenjäger, M., Ritter, A., and Obermayer, K. (1999). Active learning in self-organizing maps. In *Kohonen Maps*, pages 57–70. Referred to on pages: 105

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: data mining, inference and prediction*. Springer series in statistics. Springer, New York, N.Y. Referred to on pages: 18, 22

[Haussler, 1990] Haussler, D. (1990). Probably approximately correct learning. In *AAAI*, pages 1101–1108. Referred to on pages: 108

[Haykin, 1999] Haykin, S. S. (1999). *Neural Networks, a comprehensive foundation*. Prentice-Hall. Referred to on pages: 19

[Hettich et al., 1998] Hettich, S., Blake, C. L., and Merz, C. J. (1998). UCI repository of machine learning databases. http://www.ics.uci.edu/˜mlearn/MLRepository.html. Referred to on pages: 35, 44, 80, 104, 115, 127

[Hochbaum and Shmoys, 1985] Hochbaum, D. and Shmoys, D. (1985). A best possible heuristic for the k-centner problem. *Mathematics of Operations Research*, 10(2):180–184. Referred to on pages: 15, 22, 49

[Jaakkola et al., 1999] Jaakkola, T., Meila, M., and Jebara, T. (1999). Maximum entropy discrimination. In *Advances Neural Information Processing Systems*, volume 12, pages 470–477. Referred to on pages: 55, 122

[Japkowicz, 1999] Japkowicz, N. (1999). *Concept-learning in the absence of counter-examples: an autoassociation-based approach to classification*. PhD thesis, New Brunswick Rutgers, The State University of New Jersey. Referred to on pages: 15

[Jiang et al., 2001] Jiang, M. F., Tseng, S. S., and Su, C. M. (2001). Two-phase clustering process for outliers detection. *Pattern Recognition Letters*, 22(6-7):691–700. Referred to on pages: 15, 19, 49

[John, 1948] John, F. (1948). Extreme problems with inequalities as subsidiary conditions. *Wiley Interscience, New York*, pages 187–204. Referred to on pages: 25

[Jolliffe, 1986] Jolliffe, I. T. (1986). *Principal Component Analysis*. Springer-Verlag, New York. Referred to on pages: 20

[Juszczak et al., 2005] Juszczak, P., de Ridder, D., Tax, D. M. J., and Duin, R. P. W. (2005). Active learning by investigating unlabeled object sets. *submitterd to Machine Learning*. Referred to on pages: 95, 116

[Juszczak and Duin, 2003] Juszczak, P. and Duin, R. P. W. (2003). Uncertainty sampling for one-class classifiers. In Chawla, N., Japkowicz, N., and Kolcz, A., editors, *ICML-2003 Workshop: Learning with Imbalanced Data Sets II*, pages 81–88. Referred to on pages: 3, 4

[Juszczak and Duin, 2004] Juszczak, P. and Duin, R. P. W. (2004). Selective sampling based on the variation in label assignments. In Kittler, J., Petrou, M., and Nixon, M., editors, *Proceedings of 17th International Conference on Pattern Recognition*, volume 2, pages 375–378. IEEE Computer Society, Los Alamitos. Referred to on pages: 92

[Juszczak and Duin, 2005] Juszczak, P. and Duin, R. P. W. (2005). Learning from a test set. In *Proceedings of 4th International Conference on Computer Recognition Systems*, pages 203–210. LNCS, Springer Verlag. Referred to on pages: 121

[Khachiyan, 1996] Khachiyan, L. (1996). Rounding of polytopes in the real number model of computation. *Mathematical Operetion Research*, 21(2):307–320. Referred to on pages: 26

[Khachiyan and Todd, 1993] Khachiyan, L. and Todd, M. (1993). On the complexity of approximating the maximal inscribed ellipsoid for a polytope. *Mathematical Programming*, 61:137–159. Referred to on pages: 25

[Knorr et al., 2000] Knorr, E., Ng, R., and Tucakov, V. (2000). Distance-based outliers: algorithms and applications. *VLDB Journal: Very Large Data Bases*, 8(3–4):237–253. Referred to on pages: 15, 19

[Kohonen, 1995] Kohonen, T. (1995). *Self-organizing maps*. Springer-Verlag, Heidelberg, Germany. Referred to on pages: 19

[Kolmogorov and Tikhomirov, 1961] Kolmogorov, A. and Tikhomirov, V. (1961). ϵ-entropy and ϵ-capacity of sets in function spaces. *Trans. of the American Mathematical Society*, 17:277–364. Referred to on pages: 22, 41

[Koppel and Schler, 2004] Koppel, M. and Schler, J. (2004). Authorship verification as a one-class classification problem. In *International Conference on Machine Learning*, pages 489–495. Referred to on pages: 4, 15

[Kosinski, 1999] Kosinski, A. (1999). A procedure for the detection of multivariate outliers. *Computational statistics & data analysis*, 2:145–161. Referred to on pages: 17

[Kruskal, 1956] Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the travelling salesman problem. *Am. Math. Soc.*, 7(1):48–50. Referred to on pages: 41

[Lanckriet et al., 2003] Lanckriet, G. R. G., El Ghaoui, L., and Jordan, M. I. (2003). Robust novelty detection with single-class MPM. In Becker, S., Thrun, S., and Obermayer, T., editors, *Advances in Neural Information Processing Systems*, volume 15. MIT Press: Cambridge, MA. Referred to on pages: 15, 23

[Lauer, 2001] Lauer, M. (2001). A mixture approach to novelty detection using training data with outliers. In *Proceedings of the 12th European Conference on Machine Learning*, pages 300–311. Referred to on pages: 18

[Lewis and Gale, 1994] Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In Croft, W. B. and van Rijsbergen, C. J., editors, *Proceedings of 17th International Conference on Research and Development in Information Retrieval*, pages 3–12, Dublin, IE. Springer Verlag, Heidelberg, DE. Referred to on pages: 92, 93, 115

[Li and Lu, 1999] Li, S. and Lu, J. (1999). Face recognition using the nearest feature line method. *Neural Networks*, 10(2):439–443. Referred to on pages: 39

[Lindenbaum et al., 2004] Lindenbaum, M., Markovitch, S., and Rusakov, D. (2004). Selective sampling for nearest neighbor classifiers. *Machine Learning*, 54(2). Referred to on pages: 105, 112, 113

[Lobo et al., 1998] Lobo, M., Vandenberghe, L., Boyd, S., and Lebret, H. (1998). Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228. Referred to on pages: 28, 63

[Löfberg, 2004] Löfberg, J. (2004). YALMIP : A toolbox for modeling and optimization in MATLAB. http://control.ee.ethz.ch/~joloef/yalmip.php. Referred to on pages: 34

[Lovász et al., 1997] Lovász, L., Kannan, R., and Simonovits, M. (1997). Random walks and an $O(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11:1–50. Referred to on pages: 58

[Lütkepohl, 1996] Lütkepohl, H. (1996). *Handbook of matrices*. John Wiley Sons. Referred to on pages: 136

[Manevitz and Yousef, 2001] Manevitz, L. and Yousef, M. (2001). One-class svm for document classification. *Journal of Machine Learning Research*, 2:139–154. Referred to on pages: 15, 23

[Markou and Singh, 2003a] Markou, M. and Singh, S. (2003a). Novelty detection: a review. part 1: statistical approaches. *Signal Processing*. Referred to on pages: 15

[Markou and Singh, 2003b] Markou, M. and Singh, S. (2003b). Novelty detection: a review. part 2: neural network based approaches. *Signal Processing*. Referred to on pages: 15, 51

[Marsland, 2001] Marsland, S. (2001). *On-line novelty detection through self-organisation, with application to inspection robots*. PhD thesis, University of Manchester. Referred to on pages: 19

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York. Referred to on pages: 95, 112

[Moya et al., 1993] Moya, M. R., Koch, M. W., and Hostetler, L. D. (1993). One-class classifier networks for target recognition applications. In *Proceedings world congress on neural networks*, pages 797–801, Portland, OR. International Neural Network Society. Referred to on pages: 3, 4, 15

[Muller, 1959] Muller, M. E. (1959). A note on a method for generating points uniformly on N-dimensional spheres. *Communications of the ACM*, 2(4). Referred to on pages: 72

[Nesterov and Nemirovskii, 1994] Nesterov, Y. and Nemirovskii, A. (1994). Interior-point polynomial algorithms in convex programming. In *SIAM*. Referred to on pages: 25, 27

[Nguyen and Smeulders, 2004] Nguyen, H. T. and Smeulders, A. W. M. (2004). Active learning using pre-clustering. In *Proceedings of 21st International Conference on Machine Learning*, pages 623–630. Referred to on pages: 95

[Nunez-Garcia et al., 2003] Nunez-Garcia, J., Kutalik, Z., Cho, K.-H., and Wolkenhauer, O. (2003). Level sets and minimum volume sets of probability density functions. *Journal of Approximate Reasoning*, 34(1):25–47. Referred to on pages: 18

[Park, 2004] Park, J. M. (2004). Convergence and application of online active sampling using orthogonal pillar vectors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1197–1207. Referred to on pages: 112, 113

[Parra et al., 1996] Parra, L., Deco, G., and Miesbach, S. (1996). Statistical independence and novelty detection with information preserving nonlinear maps. *Neural Computation*, 8:260–269. Referred to on pages: 15, 49

[Parzen, 1962] Parzen, E. (1962). On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076. Referred to on pages: 15, 18, 65, 97, 106, 121

[Pękalska et al., 2003] Pękalska, E., Tax, D. M. J., and Duin, R. P. W. (2003). One-class LP classifier for dissimilarity representations. In *Neural Information Processing Systems*, pages 761–768. Referred to on pages: 15, 24

[Prim, 1957] Prim, R. C. (1957). Shortest connection networks and some generalisations. *Bell Systems Technical Journal*, pages 1389–1410. Referred to on pages: 41

[Rätsch et al., 2002] Rätsch, G., Mika, S., Schölkopf, B., and Müller, K. R. (2002). Constructing boosting algorithms from SVMs: an application to one-class classification. *IEEE PAMI*, 24(9):1184–1199. Referred to on pages: 23

[Roberts et al., 2001] Roberts, S., Holmes, C., and Denison, D. (2001). Minimum entropy data partitioning using reversible jump markov chain monte carlo. *PAMI*, 23(8):909–914. Referred to on pages: 122

[Rousseeuw and van Driessen, 1999] Rousseeuw, P. J. and van Driessen, K. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223. Referred to on pages: 5, 17, 32

[Roweis and Saul, 2000] Roweis, S. and Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326. Referred to on pages: 40

[Roy and McCallum, 2001] 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*, pages 441–448. Morgan Kaufmann, San Francisco, CA. Referred to on pages: 92, 93, 105

[Rustagi, 1994] Rustagi, J. S. (1994). *Optimization techniques in statistics*. Academic Press. Referred to on pages: 110, 136, 138

[Sain et al., 1999] Sain, S. R., Gray, H. L., Woodward, W. A., and Fisk, M. D. (1999). Outlier detection from a mixture distribution when training data are unlabeled. *Bulletin of the Seismological Society of America*, 89:294–304. Referred to on pages: 15, 18

[Sánchez et al., 1997] Sánchez, J. S., Pla, F., and Ferri, F. J. (1997). Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recognition Letters*, 18(6):507–513. Referred to on pages: 96

[Schohn and Cohn, 2000] Schohn, G. and Cohn, D. (2000). Less is more: Active learning with support vector machines. In *Proceedings of the International Conference on Machine Learning*, pages 839–846. Morgan Kaufmann, San Francisco, CA. Referred to on pages: 105

[Schölkopf et al., 2000a] Schölkopf, B., C., W. R., Smola, A. J., Shawe-Taylor, J., and Platt, J. C. (2000a). Support vector method for novelty detection. In *Neural Information Processing Systems*. Referred to on pages: 16, 49

[Schölkopf et al., 1998] Schölkopf, B., Mika, S., Smola, A. J., Rätsch, G., and Müller, K. R. (1998). Kernel PCA pattern reconstruction via approximate pre-images. In Niklasson, L., Bodén, M., and Ziemke, T., editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, pages 147 – 152, Berlin. Springer Verlag. Referred to on pages: 40

[Schölkopf et al., 2001] Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, J. A., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471. Referred to on pages: 23

[Schölkopf et al., 2000b] Schölkopf, B., Smola, A., Williamson, R. C., and Bartlett, P. L. (2000b). New support vector algorithms. *Neural Computation*, 12:1207–1245. Referred to on pages: 31

[Seeger, 2004] Seeger, M. (2004). Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(2):1–38. Referred to on pages: 105

[Seung et al., 1992] Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *Conference on Learning Theory*, pages 287–294. Referred to on pages: 92, 95

[Shyu et al., 2003] Shyu, M. L., Chen, S. C., Sarinnapakorn, K., and Chang, L. (2003). A novel anomaly detection scheme based on principal component classifier. In *IEEE Foundations and New Directions of Data Mining Workshop*, pages 172–179. Referred to on pages: 20

[Sturm, 1999] Sturm, J. F. (1999). Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 12(11):625–653. Referred to on pages: 34

[Surace et al., 1997] Surace, C., Worden, K., and Tomlinson, G. (1997). A novelty detection approach to diagnose damage in a cracked beam. In *Proceedings of SPIE*, pages 947–943. Referred to on pages: 20

[Szummer and Jaakkola, 2001] Szummer, M. and Jaakkola, T. (2001). Partially labeled classification with markov random walks. In *Neural Information Processing Systems*, volume 14, pages 945–952. Referred to on pages: 122, 123

[Tarassenko et al., 1995] Tarassenko, L., Hayton, P., and Brady, M. (1995). Novelty detection for the identification of masses in mammograms. In *International Conference on Artificial Neural Networks*, pages 442–447. Referred to on pages: 4, 15

[Tax, 2001] Tax, D. M. J. (2001). *One-class classification*. PhD thesis, Delft University of Technology. Referred to on pages: 3, 4, 7, 20, 22, 58

[Tax and Duin, 1999] Tax, D. M. J. and Duin, R. P. W. (1999). Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199. Referred to on pages: 15, 16, 49, 53, 69

[Tax and Duin, 2001] Tax, D. M. J. and Duin, R. P. W. (2001). Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*, pages 155–173. Referred to on pages: 49, 51, 53, 55, 57, 58, 61, 62, 71, 80, 83

[Tax and Duin, 2002] Tax, D. M. J. and Duin, R. P. W. (2002). Using two-class classifiers for multiclass classification. In *International Conference on Pattern Recognition*. Referred to on pages: 93

[Tax and Duin, 2004] Tax, D. M. J. and Duin, R. P. W. (2004). Support vector data description. *Machine Learning*, 54(1):45–56. Referred to on pages: 22

[Tax et al., 2006] Tax, D. M. J., Juszczak, P., and Duin, R. P. W. (2006). An overview and comparison of one-class classifiers. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, -:–. Referred to on pages: 15, 18

[Tax and Müller, 2004] Tax, D. M. J. and Müller, K. R. (2004). A consistency-based model selection for one-class classification. In *International Conference on Pattern Recognition*, pages 363–366. IEEE Computer Society, Los Alamitos, CA. Referred to on pages: 49, 53, 83

[Tenenbaum et al., 2000] Tenenbaum, J., de Silva, V., and Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323. Referred to on pages: 40

[Tishby and Slonim, 2000] Tishby, N. and Slonim, N. (2000). Data clustering by markovian relaxation and the information bottleneck method. In *Neural Information Processing Systems*, pages 640–646. Referred to on pages: 122

[Toh, 1999] Toh, K. (1999). Primal-dual path-following algorithms for determinant maximization problems with linear matrix inequalities. *Computational Optimisation Appllications*, 14:309–330. Referred to on pages: 26

[Tong and Koller, 2000] Tong, S. and Koller, D. (2000). Support vector machine active learning with applications to text classification. In Langley, P., editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 999–1006, Stanford, US. Morgan Kaufmann Publishers, San Francisco, US. Referred to on pages: 92, 95, 105, 112

[Valiant, 1984] Valiant, L. G. (1984). A theory of the learnable. *Commun. ACM*, 27(11):1134–1142. Referred to on pages: 108

[Vandenberghe et al., 1998] Vandenberghe, L., Boyd, S., and S., W. (1998). Determinant maximization with linear matrix inequality constraints. *SIAM Journal Matrix Analalisys Applications*, 19(2):499–533. Referred to on pages: 26

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley. Referred to on pages: 16, 22, 55, 121, 122

[Wolpert and Macready, 1997] Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82. Referred to on pages: 105

[Yeung and Chow, 2002] Yeung, D. Y. and Chow, C. (2002). Parzen-window network intrusion detectors. In *Proceedings of the Sixteenth International Conference on Pattern Recognition*, volume 4, pages 385–388. Referred to on pages: 18

[Yeung and Ding, 2003] Yeung, D. Y. and Ding, Y. (2003). Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition*, 36(1):229–243. Referred to on pages: 18

[Ypma, 2001] Ypma, A. (2001). *Learning methods for machine vibration analysis and health monitoring*. PhD thesis, Delft University of Technology. Referred to on pages: 80, 82

[Ypma and Duin, 1998] Ypma, A. and Duin, R. (1998). Support objects for domain approximation. In *International Conference on Artificial Neural Networks*, pages 719–724. Springer, Berlin. Referred to on pages: 4, 15, 22

[Zhang, 1998] Zhang, Y. (1998). An interior-point algorithm for the maximum-volume ellipsoid problem. Technical Report TR98-15, Department of Computational and Applied Mathematics, Rice University, Houston, TX. Referred to on pages: 26

[Zhang and Gao, 2003] Zhang, Y. and Gao, L. (2003). On numerical solution of the maximum volume ellipsoid problem. *SIAM Journal of Optimisation*, 14(1):53–76. Referred to on pages: 26